UNIVERSIDADE FEDERAL DO PARANÁ
DEPARTAMENTO DE MATEMÁTICA

Análise de Erro da Transformação de Householder Generalizada*

Autor: FEDOR PISNITCHENKO

Orientadores: JIN YUN YUAN
RICARDO BILOTI

Dissertação apresentada ao programa de Pós-Graduação em Matemática Aplicada, da Universidade Federal do Paraná, como parte dos pré-requisitos para obtenção do título de Mestre em Matemática Aplicada.

Março de 2004

*Este trabalho contou com apoio financeiro da CAPES.

Ministério da Educação
Universidade Federal do Paraná
Setor de Ciências Exatas/Departamento de Matemática
Programa de Pós-Graduação em Matemática Aplicada - PPGMAT

ATA DA 3ª DEFESA DE DISSERTAÇÃO DE MESTRADO

Ao primeiro dia do mês de março de 2004, no Anfiteatro B - Prédio PC/ET, Universidade Federal do Paraná, foi instalada pela Professora Liliana Madalena Gramani Cumin, Vice-Coordenadora do PGMAT - Programa de Pós-Graduação em Matemática Aplicada, a Banca Examinadora para a Terceira Dissertação de Mestrado em Matemática Aplicada. Estiveram presentes ao Ato, além do Coordenador do Programa de Pós-Graduação em Matemática Aplicada, professores, alunos e visitantes.

A banca examinadora, homologada pelo Colegiado do Programa de Pós-Graduação em Matemática Aplicada, ficou constituída pelos professores: Dr. Mário Cesar Zambaldi, da UFSC; Dr. Luiz Carlos Matioli, do Departamento de Matemática da UFPR; Dr. José Walter Cárdenas Sotil, do Laboratório Nacional de Computação Científica/LNCC; e Dr. Ricardo Caetano Azevedo Biloti, do Departamento de Matemática da UFPR, co-orientador da dissertação a quem coube a presidência dos trabalhos.

Às quatorze horas, a banca iniciou seus trabalhos, convidando o candidato **Fedor Pisnitchenko** a fazer a apresentação do tema da dissertação intitulada "Análise de Erro da Transformação de Householder Generalizada". Encerrada a apresentação, iniciou-se a fase de argüição pelos membros participantes. Após a argüição, a banca com pelo menos 03 (três) membros, reuniu-se para apreciação do desempenho do pós-graduando.

A banca considerou que o pós-graduando fez uma apresentação com a necessária concisão. A Dissertação apresenta contribuição à área de estudos e não foram registrados problemas fundamentais de estrutura e redação, resultando em plena e satisfatória compreensão dos objetivos pretendidos.

Tendo em vista a dissertação e a argüição, os membros presentes da banca decidiram pela sua aprovação.

Curitiba, 1º de março de 2004.

Prof. Ricardo Caetano Azevedo Biloti
Presidente

Prof. Mário Cesar Zambaldi

Prof. Luiz Carlos Matioli

Prof. José Walter Cárdenas Sotil

Ministério da Educação
UNIVERSIDADE FEDERAL DO PARANÁ
Setor de Ciências Exatas
PÓS-GRADUAÇÃO EM MATEMÁTICA APLICADA

PARECER DA BANCA EXAMINADORA

Aprovado, mediante as correções e sugestões feitas pela Banca Examinadora.

Curitiba, 1º de março de 2004.

Prof. Ricardo Caetano Azevedo Biloti
Presidente

Prof. Mário César Zambaldi

Prof. Luiz Carlos Matioli

Prof. José Walter Cárdenas Sotil

Agradecimentos

Meu sinceros agradecimentos ao meu orientador professor Jin Yun Yuan e ao meu co-orientador professor Ricardo Biloti.

Resumo

A transformação de Householder generalizada foi introduzida por LaBudde em 1963. A utilidade dessa transformação é poder reduzir uma matriz arbitrária a uma forma tridiagonal semelhante. Sua forma otimizada, chamada de transformação de Householder generalizada ótima, introduzida por Golub, Yuan, Biloti e Ramos, minimiza o número de condição da transformação e possibilita a implementação do algoritmo de tridiagonalização.

Nessa dissertação apresentamos a análise de erro da transformação de Householder generalizada ótima e do processo da tridiagonalização.

Abstract

The generalized Householder transformation, estabilished by LaBudde in 1963, is employed to reduce every given matrix to a similar tridiagonal matrix. Golub, Yuan, Biloti and Ramos determined the optimal generalized Householder transformation concerning its condition number and proposed a tridiagonalization algorithm.

In this work we develop the error analysis for the otimal generalized Householder transformation and the associated tridiagonalization process.

Sumário

Capítulo 1

Introdução

Os erros em cálculos computacionais matemáticos ocorrem por várias razões:

- a descrição do problema matemático é incorreta, em particular, os dados iniciais com erros (por exemplo, de medição);
- os métodos usados para resolução do problema não são precisos, por exemplo, para obter o resultado exato de algum problema é preciso realizar um número infinito ou não aceitável de operações aritméticas, por isso, em vez da solução exata o método obtém a solução aproximada;
- durante a entrada dos dados, operações aritméticas e saída dos dados ocorrem erros de arredondamento.

O erro de arredondamento depende de vários parâmetros: a representação dos números na máquina; tipo de aritmética implementada; como é realizado o próprio arredondamento, etc. Dependendo do computador, o erro para o mesmo procedimento aritmético pode ser maior ou menor (para maiores detalhes ver [1]).

O primeiro estudo completo sobre o efeito do erro de arredondamento nos algoritmos numéricos foi feito pelo Wilkinson nos livros *Rounding Errors in Algebraic Processes* [12] e *The Algebraic Eigenvalue Problem* [13], publicados em 1963 e 1965, respectivamente, e que se tornaram os livros clássicos da Análise Numérica e principalmente da Análise de Erro.

Recentemente, em 2002, foi publicado a segunda edição do livro de Nicholas Higham, *Accurancy and Stability of Numerical Algorithms* [6], onde é dado um tratamento moderno ao comportamento dos algoritmos numéricos em aritmética computacional.

O objetivo desta dissertação é realizar análise de erro da transformação de Householder generalizada ótima e do algoritmo de tridiagonalização introduzidos por Golub, Yuan, Biloti e Ramos em [4].

Análise de erro em problema de autovalores

O problema de autovalores é um dos assuntos mais importantes no cálculo matricial. Apesar de teoricamente não ser complexo, numericamente encontram-se muitas dificuldades para resolvê-lo. Existem muitas maneiras de calcular autovalores de uma matriz. Um dos procedimentos mais utilizados é aplicar uma seqüência de transformações similares (por exemplo, eliminação de Gauss, rotações de Givens ou transformação de Householder) para reduzir a matriz dada a uma matriz de Hesseneberg superior, para qual os autovalores são determinados mais facilmente, por meio de métodos iterativos (por exemplo, o método QR).

Em um caso especial, quando a matriz é simétrica, a transformação de Householder pode ser utilizada para tridiagonalizar a matriz. A vantagem desse método está em sua boa estabilidade numérica, que é uma das importantes propriedades da transformação de Householder mostrada por Wilkinson [13]. Além de transformação de Householder a tridiagonalização de uma matriz simétrica pode ser feita utilizando método de Lanczos e rotação de Givens entre outros.

Em 1963, LaBudde [10] introduziu uma nova transformação, que essencialmente é uma generalização da transformação de Householder, e que possibilita reduzir qualquer matriz real ou complexa a uma forma tridiagonal semelhante. Mas a desvantagem dessa transformação é que, a menos de um caso particular, quando a transformação é reduzida a transformação de Householder, ela não possui as propriedades de simetria e ortogonalidade.

Em 2003, Golub, Yuan, Biloti e Ramos em [4] estudaram as propriedades espectrais, os autovalores e valores singulares da transformação de Householder generalizada e introduziram a transformação de Householder generalizada ótima. Utilizando essa transformação, propuseram um algoritmo de tridiagonalização.

Plano de apresentação da dissertação

No Capítulo 2 expomos a base teórica para Análise de Erro. Primeiramente, introduzimos o conceito de sistema de ponto flutuante e definimos a aritmética utilizada. Depois

discutimos os conceitos de erro direto e erro reverso. A seguir apresentamos os erros das importantes operações da Análise Numérica e alguns lemas que facilitam a manipulação das expressões do erro. Terminamos o capítulo com a análise de erro da transformação de Householder. A finalidade dessa análise é, em primeiro lugar, apresentar uma análise relativamente simples e bem conhecida utilizando o conteúdo teórico exposto anteriormente, e, em segundo lugar, obter os resultados explícitos para depois compará-los com caso generalizado.

No Capítulo 3 definimos a transformação de Householder generalizada. Expomos os algoritmos introduzidos por Golub, Yuan, Biloti e Ramos e realizamos a análise de erro desses algoritmos.

Concluímos com comparações entre os erros da transformação de Housholder generalizada ótima e da transformação de Householder (obtido no Capitulo 2).

No Apêndice A falamos sobre o padrão IEEE-754, o qual especifica o sistema binário de ponto flutuante tomado como padrão para computadores a partir de 1985. No Apêndice B definimos as normas vetoriais e matriciais e expomos algumas propriedades importantes de normas utilizadas nos Capítulos 2 e 3.

Capítulo 2

Análise de erro

2.1 Sistemas numéricos

No computador cada número é armazenado numa posição de memória de tamanho fixo. Uma questão importante do projeto da arquitetura de um computador é como serão utilizados os dígitos usados para representar os números que compõe os sistemas numéricos.

Existem dois sistemas numéricos que são usados nos computadores digitais, o sistema de ponto fixo e o sistema de ponto flutuante. Cada um destes sistemas requer seu próprio conceito de aritmética computacional. No nosso estudo trabalharemos apenas com a aritmética de ponto flutuante, que é a usada de fato para realizar cálculos científicos.

2.1.1 Sistema numérico de ponto flutuante

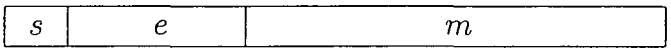
O sistema numérico de ponto flutuante, denotado por F, é caracterizado pelos seguintes parâmetros:

- *base de máquina* β,
- *precisão* t e
- *extensão exponencial* $e_{min} \leq e \leq e_{max}$.

Um elemento de F é representado na forma

$$y = \pm m \times \beta^{e-t}, \tag{2.1}$$

onde a *mantissa* m é um número inteiro satisfazendo $0 \leq m \leq \beta^t - 1$. Na memória do computador o número é representado como mostra a figura abaixo, onde s representa o sinal.



Para garantir unicidade de representação para cada $y \in F$ assumimos que $m \geq \beta^{t-1}$ se $y \neq 0$. A faixa dos números de ponto flutuante não nulos em F está contida em $[\beta^{e_{min}-1}, \beta^{e_{max}}(1 - \beta^{-t})]$.

Uma outra maneira para representar um elemento de F é

$$y = \pm\beta^e \left(\frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \cdots + \frac{d_t}{\beta^t} \right) = \pm .d_1 d_2 \ldots d_t \times \beta^e,$$

onde cada digito d_i satisfaz $0 \leq d_i \leq \beta - 1$, e para números normalizados $d_1 \neq 0$. Preferimos a representação mais concisa (2.1), com a qual é mais simples de trabalhar.

Agora vamos ver como são feitas as operações aritméticas com números de ponto flutuante.

2.1.2 Aritmética de ponto flutuante

Definição 2.1. Seja $G \subset \mathbb{R}$ o conjunto de todos os números da forma (2.1) sem restrição no expoente e. Se $x \in \mathbb{R}$ então $fl(x)$ denota um elemento de G mais próximo a x e a transformação $x \mapsto fl(x)$ é chamada de *arredondamento*.

Dizemos que $fl(x)$ produz resultado *overflow* se $|fl(x)| > \max\{|y| : y \in F\}$ e *underflow* se $0 < |fl(x)| < \min\{|y| : 0 \neq y \in F\}$.

O resultado a seguir mostra que todo número real x tal que $\min\{|y| : y \in F\} < |x| < \max\{|y| : y \in F\}$ pode ser aproximado por um elemento de F com erro relativo menor que

$$u = \frac{1}{2}\beta^{1-t}. \tag{2.2}$$

A quantidade u é chamada de *unidade de arredondamento*.

Teorema 2.1. *Seja* $x \in \mathbb{R}$ *tal que* $\min\{|y| : y \in F\} < |x| < \max\{|y| : y \in F\}$ *então*

$$fl(x) = x(1 + \delta), \qquad |\delta| \leq u. \tag{2.3}$$

Demonstração. Podemos assumir, sem perda de generalidade, que $x > 0$. Representando x da seguinte maneira

$$x = \mu \times \beta^{e-t}, \text{ com } \beta^{t-1} \leq \mu \leq \beta^t - 1,$$

vemos que x está entre dois números do ponto flutuante $y_1 = m \times \beta^{e-t}$ e $y_2 = M \times \beta^{e-t}$, onde $m \leq \mu \leq M$, ou seja, $fl(x) = y_1$ ou $fl(x) = y_2$. Logo

$$|fl(x) - x| \leq \frac{|y_2 - y_1|}{2} \leq \frac{|M - m| \times \beta^{e-t}}{2} \leq \frac{\beta^{e-t}}{2}.$$

Portanto

$$|\delta| = \left|\frac{fl(x) - x}{x}\right| \leq \frac{\frac{1}{2}\beta^{e-t}}{\mu \times \beta^{e-t}} \leq \frac{1}{2}\beta^{1-t} = u.$$

□

O Teorema 2.1 afirma que $fl(x)$ é igual a x vezes um fator muito próximo de 1. A representação $1 + \delta$ para esse fator é uma escolha comum, mas não é a única, como mostrado no teorema a seguir.

Teorema 2.2. *Seja* $x \in \mathbb{R}$ *tal que* $\min\{|y| : y \in F\} < |x| < \max\{|y| : y \in F\}$ *então*

$$fl(x) = \frac{x}{1+\delta}, \qquad |\delta| \leq u.$$

Demonstração. Seja x definido como no Teorema 2.1, relembrando que $y_1 \leq fl(x) \leq y_2$ e $m \geq \beta^{t-1}$ temos

$$|\delta| = \left|\frac{x - fl(x)}{fl(x)}\right| \leq \frac{\frac{1}{2}\beta^{e-t}}{m \times \beta^{e-t}} \leq \frac{1}{2}\beta^{1-t} = u$$

□

Definição 2.2. Sejam $x, y \in F$ dois números de ponto flutuante e " op " qualquer uma das quatro operações aritméticas ($+$, $-$, $\times$, $\div$). Então o resultado computacional de $(x \text{ op } y)$ é dado pelo seguinte modelo

$$fl(x \text{ op } y) = (x \text{ op } y)(1 + \delta), \qquad |\delta| \leq u. \tag{2.4}$$

Esse modelo diz que o valor obtido de $(x \text{ op } y)$ pela máquina é tão bom quanto arredondar a resposta exata[1], ou seja, o erro relativo é menor ou igual a unidade de

[1]Existem algumas máquinas cujas operações aditivas do ponto flutuante satisfazem outro modelo aritmético dado por $fl(x \pm y) = x(1 + \alpha) \pm y(1 + \beta)$ onde $|\alpha|,|\beta| \leq u$ (ver Higham (1996, pág. 48-50)).

arredondamento u. Também consideraremos o mesmo resultado para operação raiz quadrada

$$fl(\sqrt{x}) = \sqrt{x}(1+\delta), \qquad |\delta| \leq u. \tag{2.5}$$

O padrão IEEE-754 (Apêndice A), seguido pela maioria dos processadores modernos da familia x86 (ver [8]), satisfaz esse modelo aritmético, inclusive para extração de raiz quadrada. Mais detalhes sobre padrão IEEE-754 e aritmética do ponto flutuante são dados em [7],[2] e [11].

Usando Teorema 2.2 podemos escrever (2.4) e (2.5) de outra maneira

$$fl(x \text{ op } y) = \frac{(x \text{ op } y)}{(1+\delta)}, \qquad |\delta| \leq u, \tag{2.6}$$

e

$$fl(\sqrt{x}) = \frac{\sqrt{x}}{(1+\delta)}, \qquad |\delta| \leq u. \tag{2.7}$$

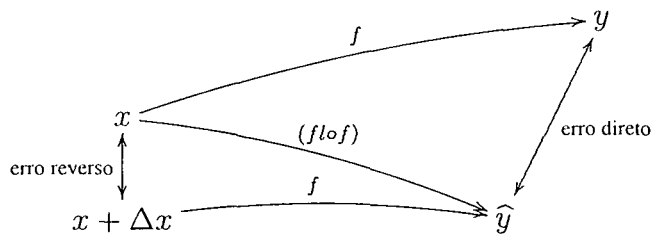
2.2 Erro direto e erro reverso

Suponhamos que $\widehat{y}$ é a aproximação de $y = f(x)$ na aritmética de ponto flutuante, onde f é uma função real com variáveis reais. Formularemos a seguinte pergunta: como deveremos avaliar a "qualidade" da aproximação de $\widehat{y}$? A resposta mais evidente é avaliar o erro absoluto ($|y - \widehat{y}|$) ou erro relativo ($|y - \widehat{y}|/|y|$) da solução $\widehat{y}$. Mas também existe uma outra possibilidade. Em vez de concentrar-se no erro de $\widehat{y}$, podemos procurar Δx tal que

$$\widehat{y} = f(x + \Delta x). \tag{2.8}$$

Em geral, podem existir muitos valores Δx satisfazendo (2.8), então devemos procurar o menor deles.

Chamamos $|\Delta x|$ (ou $\min |\Delta x|$) de erro reverso (*backward error*) e $|y - \widehat{y}|$ de erro direto (*forward error*).



O algoritmo para calcular $y = f(x)$ chama-se reversamente estável (*backward stable*) se, para todo x, o erro reverso do resultado $\widehat{y}$ é pequeno, ou seja, se $\widehat{y} = f(x + \Delta x)$ para algum Δx pequeno (a definição de "pequeno" depende do contexto). E, analogamente, o algoritmo chama-se estável (*forward stable*), quando o erro direto é pequeno.

2.3 Cancelamento catastrófico

Cancelamento catastrófico é um fenômeno de perda de dígitos significativos que ocorre quando números pequenos são obtidos pela subtração de números grandes. Como exemplo calcularemos a menor raiz da equação $y^2 - 140y + 1$, com base $\beta = 10$ e a precisão da mantissa $t = 4$. Temos

$$y_1 = \frac{140 - \sqrt{140^2 - 4}}{2},$$

de onde

$$y_1 = 70 - \sqrt{4899}.$$

Agora $\sqrt{4899} = 69.992857...$, arredondando a mantissa para precisão $t = 4$ temos

$$\widehat{y}_1 = 70 - 69.99 = 0.01.$$

Substituindo o valor de $\widehat{y}_1$ obtido na equação original

$$0.01^2 - 140 * 0.01 + 1 = -0.3999$$

vemos que o resultado está longe de 0. Para melhorar a aproximação $\widehat{y}_1$ basta evitar a subtração

$$y_1 = 70 - \sqrt{4899} = \frac{(70 - \sqrt{4899})(70 + \sqrt{4899})}{70 + \sqrt{4899}} = \frac{1}{70 + \sqrt{4899}}.$$

Realizando os cálculos a partir dessa expressão temos

$$70 + \sqrt{4889} \approx 140.0, \qquad \frac{1}{140.0} = 0.00714285...,$$

portanto

$$\widehat{y}_1 = 0.007143.$$

Substituindo novamente o valor de $\widehat{y}_1$ obtido na equação original

$$0.007143^2 - 140 * 0.007143 + 1 = 3.102244 * 10^{-5}$$

vemos que $\widehat{y}_1 = 0.007143$ está muito próximo a raiz exata.

Nas próximas seções calcularemos o erro cometido em operações comuns em Álgebra Linear Numérica, que serão úteis pata a análise que se segue.

2.4 Produto interno

Consideremos o produto interno $s_n = x^T y$, onde $x, y \in \mathbb{R}^n$, calculado pelo seguinte algoritmo

Algoritmo 2.1. **Produto Interno entre** x **e** y

1. $s = 0$
2. *Para* $k = 1 : n$
   2.1. $s = s + x_k y_k$

Denotemos a p-ésima soma $\sum_{i=1}^{p} x_i y_i$ por s_p e seja $\widehat{s}_p = fl(s_p)$. Então $\widehat{s}_1 = x_1 y_1(1 + \delta_1)$ com $|\delta_1| \leq u$ e para $p = 2 : n$,

$$\widehat{s}_p = fl(\widehat{s}_{p-1} + x_p y_p) = (\widehat{s}_{p-1} + x_p y_p(1 + \delta_p))(1 + \epsilon_p), \qquad |\delta_p|, |\epsilon_p| \leq u.$$

Ou seja, $\widehat{s}_2$ é dado por

$$\begin{aligned} \widehat{s}_2 &= [\widehat{s}_1 + x_2 y_2(1 + \delta_2)](1 + \epsilon_2) \\ &= x_1 y_1(1 + \delta_1)(1 + \epsilon_2) + x_2 y_2(1 + \delta_2)(1 + \epsilon_2). \end{aligned}$$

A próxima soma, $\widehat{s}_3$, é dada por

$$\begin{aligned} \widehat{s}_3 &= [\widehat{s}_2 + x_3 y_3(1 + \delta_3)](1 + \epsilon_3) \\ &= [x_1 y_1(1 + \delta_1)(1 + \epsilon_2) + x_2 y_2(1 + \delta_2)(1 + \epsilon_2) + x_3 y_3](1 + \epsilon_3) \\ &= x_1 y_1(1 + \delta_1) \prod_{k=2}^{3}(1 + \epsilon_k) + x_2 y_2(1 + \delta_2) \prod_{k=2}^{3}(1 + \epsilon_k) + x_3 y_3(1 + \delta_3)(1 + \epsilon_3). \end{aligned}$$

Prosseguindo dessa forma, obtemos a expressão de $\widehat{s}_n$

$$\widehat{s}_n = x_1 y_1(1 + \delta_1) \prod_{k=2}^{n}(1 + \epsilon_k) + x_2 y_2(1 + \delta_2) \prod_{k=2}^{n}(1 + \epsilon_k) + \ldots + x_n y_n(1 + \delta_n)(1 + \epsilon_n),$$

O resultado computacional de produto interno, usando a última expressão, pode ser representado na seguinte forma

$$fl(x^T y) = \widehat{s}_n = \sum_{i=1}^{n} x_i y_i (1 + \varepsilon_i)$$

onde

$$(1 + \varepsilon_i) = (1 + \delta_i) \prod_{j=i}^{n} (1 + \epsilon_j) \tag{2.9}$$

com $\epsilon_1 = 0$. O próximo Lema mostra como essa expressão pode ser simplificada.

Definição 2.3. Seja $\Gamma : \{x \in \mathbb{R} : 0 \leq xu < 1\} \rightarrow \mathbb{R}$ uma função real dada por

$$\Gamma(x) = \frac{xu}{1 - xu}. \tag{2.10}$$

Definimos $\Theta[x]$ como uma quantidade numérica tal que

$$|\Theta[x]| \leq \Gamma(x). \tag{2.11}$$

Lema 2.3. *Se* $|\delta_i| \leq u$, $\rho_i = \pm 1$ *para* $i = 1 : n$, *e* $nu < 1$, *temos que*

$$\prod_{i=1}^{n} (1 + \delta_i)^{\rho_i} = 1 + \Theta[n].$$

Observação: Γ e Θ definidos por (2.10) e (2.11), respectivamente, serão usados em todos os cálculos futuros que envolvem produto interno.

Demonstração. Se ρ_i é positivo para todo i, ou seja

$$\prod_{i=1}^{n} (1 + \delta_i)^{\rho_i} = \prod_{i=1}^{n} (1 + \delta_i),$$

então

$$\left| \prod_{i=1}^{n} (1 + \delta_i)^{\rho_i} - 1 \right| = \left| \prod_{i=1}^{n} (1 + \delta_i) - 1 \right| \leq (1 + u)^n - 1.$$

Assim como $1 + x \leq e^x$ para $x \geq 0$, temos que $(1 + u)^n < e^{nu}$. Expandindo e^{nu} em serie de Taylor e lembrando que $nu < 1$, obtemos

$$\begin{aligned}
(1 + u)^n - 1 &< nu + \frac{(nu)^2}{2!} + \frac{(nu)^3}{3!} + \ldots \\
&< nu \left(1 + \frac{nu}{2} + \left(\frac{nu}{2} \right)^2 + \left(\frac{nu}{2} \right)^3 + \ldots \right) \\
&= nu \frac{1}{1 - nu/2}.
\end{aligned} \tag{2.12}$$

Agora, voltando à hipótese $\rho_i = \pm 1$ para $i = 1 : n$, e notando que podemos escrever

$$\prod_{i=1}^{n}(1+\delta_i)^{\rho_i} = \frac{\prod_{i=1}^{p}(1+\delta_{l_i})}{\prod_{i=1}^{q}(1+\delta_{m_i})},$$

onde $p+q=n$, $\{l_1 \ldots l_p\} \cup \{m_1 \ldots m_q\} = \{1 \ldots n\}$ e $\{l_1 \ldots l_p\} \cap \{m_1 \ldots m_q\} = 0$, temos

$$\left|\frac{\prod_{i=1}^{p}(1+\delta_{l_i})}{\prod_{i=1}^{q}(1+\delta_{m_i})} - 1\right| \leq \left|\frac{(1+u)^p}{(1-u)^q} - 1\right|.$$

Usando relação $(1-u)^q \geq (1-qu)$ e (2.12) tem-se

$$\begin{aligned}
\left|\prod_{i=1}^{n}(1+\delta_i)^{\rho_i} - 1\right| &\leq \left|\frac{(1+u)^p}{(1-qu)} - 1\right| = \left|\frac{[(1+u)^p - 1] + qu}{(1-qu)}\right| \\
&\leq \left|\frac{[pu/(1-pu/2)] + qu)}{1-qu}\right| = \left|\frac{(p+q)u - pqu^2/2}{(1-pu/2)(1-qu)}\right| \\
&\leq \left|\frac{(p+q)u}{1-(\frac{p}{2}+q)u + pqu^2/2}\right| \leq \left|\frac{(p+q)u}{1-(\frac{p}{2}+q)u}\right| \\
&\leq \left|\frac{(p+q)u}{1-(p+q)u}\right| = \frac{nu}{1-nu}.
\end{aligned}$$

Ou seja, $|\Theta[n]| = |\prod_{i=1}^{n}(1+\delta_i)^{\rho_i} - 1| \leq \Gamma(n)$. □

Aplicando o Lema 2.3 à (2.9) obtemos o erro reverso para o cálculo do produto interno pelo algoritmo 2.1

$$fl(x^T y) = x_1y_1(1+\Theta'[n]) + x_2y_2(1+\Theta[n]) + x_3y_3(1+\Theta[n-1]) + \ldots + x_ny_n(1+\Theta[2]). \tag{2.13}$$

Definindo $|x|$ como um vetor cujos elementos são $|x_i|$, ou seja,

$$|x| = \begin{pmatrix} |x_1| \\ |x_2| \\ \vdots \\ |x_n| \end{pmatrix},$$

a representação mais concisa do erro reverso é dada por

$$fl(x^T y) = (x+\Delta x)^T y = x^T(y+\Delta y), \quad |\Delta x| \leq \Gamma(n)|x|, \quad |\Delta y| \leq \Gamma(n)|y|. \tag{2.14}$$

O majorante para o erro direto segue imediatamente de (2.14):

$$|fl(x^T y) - x^T y| \leq \Gamma(n) \sum_{i=1}^{n} |x_i y_i| = \Gamma(n)|x|^T|y| \tag{2.15}$$

Note que se $|x^T y| \ll |x|^T|y|$, então nada garante que o erro relativo em $fl(x^T y)$ seja pequeno.

Para simplificar a análise envolvendo erro relativo, quando $x^T y \neq 0$ usaremos o seguinte resultado.

Lema 2.4. *Se $x^T y \neq 0$ e $nu \leq 1$, então*

$$fl(x^T y) = x^T y \left(1 + \Theta[n] \frac{|x|^T|y|}{|x^T y|}\right) \tag{2.16}$$

Demonstração. A demonstração é imediata, segue da equação (2.15) e Lema anterior. □

2.4.1 Acumulação do produto interno

Existem computadores com possibilidade de acumular produto interno usando o dobro da precisão nos cálculos intermediários. Ou seja, se x e y são dois vetores cujos elementos são números de ponto flutuante e o tamanho de mantissa é t, então o produto possui a mantissa de $2t - 1$ ou de $2t$ dígitos, portanto pode ser represado exatamente com mantissa de $2t$ dígitos. Nessa situação o acumulo computacional do produto interno tem um erro *relativo* muito bom, isto é, $fl(x^T y) = x^T y(1 + \delta)$ onde $|\delta| \approx u$.[2]

Uma outra possibilidade para reduzir o erro é implementar um outro tipo do somatório, onde o erro não depende da dimensão do vetor. Para maiores detalhes ver [9], §4.2.2 ou [5].

2.4.2 Multiplicação matricial

Sejam $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$ e $y = Ax$. O vetor y pode ser representado como m produtos internos, $y_i = a_i^T x$, $i = 1 : m$, onde a_i^T é a i-ésima linha de A. Usando (2.14) temos

$$\widehat{y}_i = fl(a_i^T x) = (a_i + \Delta a_i)^T x, \qquad |\Delta a_i| \leq \Gamma(n)|a_i|.$$

[2]Quando não especificarmos a forma como o cálculo do produto interno é feito, consideraremos que o produto interno é acumulado pelo Algoritmo 2.1 com erro dado por (2.15).

Com isso o erro reverso é dado por

$$\widehat{y} = (A + \Delta A)x, \qquad |\Delta A| \leq \Gamma(n)|A| \tag{2.17}$$

e um majorante para o erro direto é

$$|y - \widehat{y}| \leq \Gamma(n)|A||x|. \tag{2.18}$$

O erro normalizado segue diretamente da definição de norma (Apêndice B, Sessões B.3 e B.4). Por exemplo, na norma-1 e ∞

$$\|y - \widehat{y}\|_p \leq \Gamma(n)\|A\|_p\|x\|_p, \qquad p = 1, \infty,$$

e para norma 2

$$\|y - \widehat{y}\|_2 \leq \Gamma(n)\|A\|_F\|x\|_2 \leq \sqrt{\text{posto}(A)}\Gamma(n)\|A\|_2\|x\|_2. \tag{2.19}$$

De fato,

$$\begin{aligned}
\|y - \widehat{y}\|_2 = \left(\sum_{i=1}^{m} |y_i - \widehat{y}_i|^2\right)^{1/2} &\leq \Gamma(n)\left(\sum_{i=1}^{m}(|A(i,:)||x|)^2\right)^{1/2} \\
&\leq \Gamma(n)\left(\sum_{i=1}^{m}(\|A(i,:)\|_2\|x\|_2)^2\right)^{1/2} \leq \Gamma(n)\|x\|_2\left(\sum_{i=1}^{m}(\|A(i,:)\|_2)^2\right)^{1/2} \\
&= \Gamma(n)\|x\|_2\left(\sum_{i=1}^{m}\left(\sum_{j=1}^{n}A(i,j)^2\right)\right)^{1/2} = \Gamma(n)\|x\|_2\|A\|_F \\
&\leq \sqrt{\text{posto}(A)}\Gamma(n)\|x\|_2\|A\|_2
\end{aligned}$$

Agora consideremos a multiplicação matricial: $C = AB$, onde $A \in \mathbb{R}^{m\times n}$ e $B \in \mathbb{R}^{n\times k}$. A j-ésima coluna de C é dada por $c_j = Ab_j$, onde $c_j = C(:,j)$ e $b_j = B(:,j)$. De (2.14),

$$\widehat{c}_j = (A + \Delta A_j)b_j, \qquad |\Delta A_j| \leq \Gamma(n)|A|.$$

Dai temos o majorante para erro direto

$$|C - \widehat{C}| \leq \Gamma(n)|A||B| \qquad (A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{n\times k}).$$

O majorante normalizado para norma-1, ∞ e F correspondente a

$$\|C - \widehat{C}\|_p \leq \Gamma(n)\|A\|_p\|B\|_p, \qquad p = 1, \infty, F.$$

e para norma-2 a

$$\|C - \widehat{C}\|_2 \leq \Gamma(n)\|A\|_F\|B\|_2 \leq \sqrt{\text{posto}(A)}\Gamma(n)\|A\|_2\|B\|_2.$$

2.4.3 Notação em Análise de Erro

Em análises mais complexas baseados no Lema 2.3 é necessário manipular os termos $1 + \Theta[k]$ e $\Gamma(k)$. O lema a seguir simplifica algumas das combinações mais usadas.

Lema 2.5. *Seja* $|\Theta[k]| \leq \Gamma(k) = ku/(1-ku)$ *para qualquer inteiro positivo* k. *As seguintes relações são satisfeitas*

(i) $(1+\Theta[k])(1+\Theta[j]) = 1+\Theta[k+j]$,

(ii) $\dfrac{(1+\Theta[k])}{(1+\Theta[j])} = \begin{cases} 1+\Theta[k+j] & j \leq k, \\ 1+\Theta[k+2j] & j > k, \end{cases}$

(iii) $\Gamma(k) + u \leq \Gamma(k+1)$,

(iv) $i\Gamma(k) \leq \Gamma(ik)$,

(v) $\Gamma(k) + \Gamma(j) + \Gamma(k)\Gamma(j) \leq \Gamma(k+j)$.

Demonstração. (i) é conseqüência direta de (v), pois

$$(1+\Theta[k])(1+\Theta[j]) = 1 + \Theta[k] + \Theta[j] + \Theta[k]\Theta[j]$$

onde, por hipótese

$$|\Theta[k] + \Theta[j] + \Theta[k]\Theta[j]| \leq \Gamma(k) + \Gamma(j) + \Gamma(k)\Gamma(j).$$

Para provar (iii), (iv) e (v) basta aplicar a definição:

$$\Gamma(k) + u = \frac{ku}{1-ku} + u = \frac{(k+1)u - ku^2}{1-ku} \leq \frac{(k+1)u}{1-(k+1)u} = \Gamma(k+1),$$
$$i\Gamma(k) = \frac{iku}{1-ku} \leq \frac{iku}{1-iku} = \Gamma(ik)$$

e

$$\begin{aligned} \Gamma(k) + \Gamma(j) + \Gamma(k)\Gamma(j) &= \frac{ku}{1-ku} + \frac{ju}{1-ju} + \frac{kju^2}{(1-ku)(1-ju)} \\ &= \frac{ku(1-ju) + ju(1-ku) + kju^2}{(1-ku)(1-ju)} \\ &= \frac{(k+j)u - kju^2}{1-(k+j)u+kju^2} \leq \frac{(k+j)u}{1-(k+j)u} = \Gamma(k+j). \end{aligned}$$

Para provar (ii), observemos primeiro que

$$\frac{(1+\Theta[k])}{(1+\Theta[j])} = 1 + \frac{\Theta[k]-\Theta[j]}{1+\Theta[j]}.$$

Portanto

$$\left|\frac{\Theta[k]-\Theta[j]}{1+\Theta[j]}\right| \leq \frac{\frac{ku}{1-ku}+\frac{ju}{1-ju}}{1+\frac{ju}{1-ju}} = \frac{(k+j)u-2kju^2}{(1-2ju)(1-ku)}.$$

Se $j \leq k$,

$$\frac{(k+j)u-2kju^2}{(1-2ju)(1-ku)} \leq \frac{(k+j)u}{1-(k+j)u} = \Gamma(k+j),$$

senão,

$$\begin{aligned}\frac{(k+j)u-2kju^2}{(1-2ju)(1-ku)} &\leq \frac{(k+j)u-2kju^2}{1-(2j+k)u+2kju^2} \\ &\leq \frac{(k+2j)u}{1-(k+2j)u} = \Gamma(k+2j).\end{aligned}$$

□

O próximo lema mostra como majorar o efeito de perturbação no produto matricial.

Lema 2.6. *Seja $X_j + \Delta X_j \in \mathbb{R}^{n\times n}$. Se para uma norma consistente tem-se $\|\Delta X_j\| \leq \delta_j\|X_j\|$ para todo j, então*

$$\left\|\prod_{j=0}^{m}(X_j+\Delta X_j) - \prod_{j=0}^{m} X_j\right\| \leq \left(\prod_{j=0}^{m}(1+\delta_j)-1\right)\prod_{j=0}^{m}\|X_j\|.$$

Demonstração. Procederemos por indução em m. Para $m = 0$ a condição evidentemente é satisfeita. Suponhamos que a hipótese vale para $m = k$, então

$$\begin{aligned}\left\|\prod_{j=0}^{k+1}(X_j+\Delta X_j) - \prod_{j=0}^{k+1} X_j\right\| &= \left\|(X_{k+1}+\Delta X_{k+1})\prod_{j=0}^{k}(X_j+\Delta X_j) - X_{k+1}\prod_{j=0}^{k+1} X_j\right\| \\ &\leq \|X_{k+1}\|\left\|\prod_{j=0}^{k}(X_j+\Delta X_j) - \prod_{j=0}^{k} X_j\right\| + \|\Delta X_{k+1}\|\left\|\prod_{j=0}^{k}(X_j+\Delta X_j)\right\| \\ &\leq \left(\prod_{j=0}^{k}(1+\delta_j)-1\right)\prod_{j=0}^{k+1}\|X_j\| + \delta_{k+1}\|X_{k+1}\|\left\|\prod_{j=0}^{k}(X_j+\Delta X_j)\right\|.\end{aligned}$$

Notando que $\left\|\prod_{j=0}^{k}(X_j+\Delta X_j)\right\| \leq \left\|\prod_{j=0}^{k}(X_j+\Delta X_j)-\prod_{j=0}^{k}X_j\right\| + \left\|\prod_{j=0}^{k}X_j\right\|$ e aplicando hipótese de indução temos

$$\delta_{k+1}\|X_{k+1}\|\left\|\prod_{j=0}^{k}(X_j+\Delta X_j)\right\| \leq \delta_{k+1}\left(\prod_{j=0}^{k}(1+\delta_j)\right)\prod_{j=0}^{k+1}\|X_j\|.$$

E finalmente obtemos

$$\left\|\prod_{j=0}^{k+1}(X_j+\Delta X_j)-\prod_{j=0}^{k+1}X_j\right\| \leq \left(\prod_{j=0}^{k+1}(1+\delta_j)-1\right)\prod_{j=0}^{k+1}\|X_j\|.$$

□

O próximo lema é uma variação do Lema 2.6, com uma demonstração análoga, utilizando o fato de que $\|AB\|_F \leq \|A\|_2\|B\|_F$ para quaisquer matrizes A e B.

Lema 2.7. *Seja $X_j+\Delta X_j \in \mathbb{R}^{n\times n}$. Se para uma norma consistente tem-se $\|\Delta X_j\|_F \leq \delta_j\|X_j\|_2$ para todo j, então*

$$\left\|\prod_{j=0}^{m}(X_j+\Delta X_j)-\prod_{j=0}^{m}X_j\right\|_F \leq \left(\prod_{j=0}^{m}(1+\delta_j)-1\right)\prod_{j=0}^{m}\|X_j\|_2.$$

Como exemplo, vamos agora fazer análise de erro da transformação de Householder. Nessa análise utilizaremos a maioria dos resultados teóricos expostos acima.

2.5 Transformação de Householder

Uma matriz $P \in \mathbb{R}^{n\times n}$ da forma

$$P = I - \frac{2}{v^Tv}vv^T \tag{2.20}$$

é chamada de *transformação de Householder* e possui as propriedades de simetria e ortogonalidade. Quando um vetor x é multiplicado por P, sua imagem é refletida no hiperplano $\text{span}\{v\}^\perp$. A transformação de Householder é muito usada para zerar elementos de um vetor. Em particular, dado um $x \in \mathbb{R}^n$ não nulo, é simples encontrar um vetor v em (2.20) tal que Px é um múltiplo de $e_1 = (1,0,\ldots,0)^T$. Note que aplicando P a um vetor x tem-se

$$Px = x - \left(\frac{2v^Tx}{v^Tv}\right)v, \tag{2.21}$$

ou seja, para que Px pertença a span$\{e_1\}$, devemos ter $v \in$ span$\{x, e_1\}$. Colocando $v = x + \alpha e_1$ obtemos

$$\begin{cases} v^T v = x^T x + \alpha x_1, \\ v^T x = x^T x + 2\alpha x_1 + \alpha^2, \end{cases}$$

e substituindo em (2.21)

$$Px = \left[1 - 2\frac{x^T x + \alpha x_1}{x^T x + 2\alpha x_1 + \alpha^2}\right] x - 2\alpha \frac{v^T x}{v^T v} e_1.$$

Para que o coeficiente de x seja nulo, α deve ser igual a $\pm\|x\|_2$. Ou seja, se $v = x \pm \|x\|_2 e_1$, então $Px = \mp\|x\|_2 e_1$. Para garantir a estabilidade numérica queremos que $\|v\|_2 \geq \|x\|_2$, logo escolhemos $\alpha = \text{sign}(x_1)\|x\|_2$.

Usando as observações acima podemos construir um algoritmo para determinar a matriz $P = I - \beta v v^T$, com $\beta = 2/v^T v$, tal que $Px = \alpha e_1$, onde $x \in \mathbb{R}^n$ é um vetor arbitrário.

Algoritmo 2.2. **Transformação de Householder**

1. $v = x$
2. $\alpha = 0$
3. *Para* $i = 1 : n$
   - 3.1. $\alpha = \alpha + x_i^2$
4. $\phi = sign(x_1)\sqrt{\alpha}$
5. $v_1 = v_1 + \phi$
6. $\beta = 1/(x_1\phi + \alpha)$

2.5.1 Tridiagonalização

Podemos usar as transformações de Householder para reduzir uma matriz simétrica qualquer a uma matriz tridiagonal semelhante. De fato, seja $A \in \mathbb{R}^{n\times n}$ uma matriz simétrica e $P_1 \in \mathbb{R}^{(n-1)\times(n-1)}$, a transformação de Householder tal que,

$$P_1 A(2:n, 1) = \alpha_1 e_1^{n-1},$$

onde $e_1^{n-1} = (1, 0, \ldots, 0)^T \in \mathbb{R}^{n-1}$. Se

$$Q_1 = \begin{pmatrix} 1 & 0 \\ 0 & P_1 \end{pmatrix},$$

então

$$\begin{aligned} Q_1 A Q_1 &= \begin{pmatrix} 1 & 0 \\ 0 & P_1 \end{pmatrix} \begin{pmatrix} a_{11} & A(1, 2:n) \\ A(2:n, 1) & A(2:n, 2:n) \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & P_1 \end{pmatrix} \\ &= \begin{pmatrix} a_{11} & A(1, 2:n)P_1 \\ P_1 A(2:n, 1) & P_1 A(2:n, 2:n) P_1 \end{pmatrix}, \end{aligned}$$

sendo P e A matrizes simétricas, $(A(1, 2:n)P_1)^T = P_1 A(2:n, 1) = \alpha_1 e_1^{n-1}$ e

$$Q_1 A Q_1 = \left(\begin{array}{c|ccc} a_{11} & \alpha_1 & 0 \ \ldots & 0 \\ \hline \alpha_1 & & & \\ 0 & & & \\ \vdots & & A_{[n-1]} & \\ 0 & & & \end{array}\right),$$

onde $A_{[n-1]} = P_1 A(2:n, 2:n) P_1$. Agora definindo $P_2 \in \mathbb{R}^{(n-2)\times(n-2)}$ por

$$P_2 A_{[n-1]}(3:n, 2) = \alpha_2 e_1^{n-2},$$

$e_1^{n-2} = (1, 0, \ldots, 0)^T \in \mathbb{R}^{n-2}$, e

$$Q_2 = \begin{pmatrix} I_2 & 0 \\ 0 & P_2 \end{pmatrix}$$

obtemos

$$Q_2 Q_1 A Q_1 Q_2 = \left(\begin{array}{cc|ccc} a_{11} & \alpha_1 & 0 & 0 \ \ldots & 0 \\ \alpha_1 & \tilde{a}_{22} & \alpha_2 & 0 \ \ldots & 0 \\ \hline 0 & \alpha_2 & & & \\ 0 & 0 & & & \\ \vdots & \vdots & & A_{[n-2]} & \\ 0 & 0 & & & \end{array}\right),$$

onde $A_{n-2} = P_2 A_{[n-1]}(2:n-1, 2:n-1) P_2$. Continuando dessa forma, obtemos a matriz tridiagonal T dada por

$$T = Q_{n-2} \cdots Q_1 A Q_1 \cdots Q_{n-2} \tag{2.22}$$

onde

$$Q_i = \begin{pmatrix} I_i & 0 \\ 0 & P_i \end{pmatrix} \qquad i = 1 \ldots n-2, \tag{2.23}$$

e P_i são a transformações de Householder.

2.5.2 Análise de erro da transformação de Householder

Lema 2.8. *Seja $x \in \mathbb{R}^n$. Então $\widehat{\beta}$ e $\widehat{v}$ obtidos pelo Algoritmo 2.2 satisfazem*

$$\widehat{\beta} = \beta(1 + \Theta[2n+6]), \qquad \begin{cases} \widehat{v}_1 = v_1(1 + \Theta[n+2]) \\ \widehat{v}_{2:n} = v_{2:n} \end{cases}.$$

E se para $b \in \mathbb{R}^n$, $y = Pb = b - 2\beta vv^T b$, então $\widehat{y}$ satisfaz

$$\widehat{y} = (P + \Delta P)b, \qquad \|\Delta P\|_F = \|\Delta P\|_2 \leq \Gamma(10n+27). \tag{2.24}$$

Demonstração. Seguindo os passos do Algoritmo 2.2, primeiramente, calculamos produto interno $\widehat{\alpha} = fl(x^T x) = x_1^2(1+\Theta'[n]) + x_2^2(1+\Theta[n]) + \ldots + x_n^2(1+\Theta[2]) = x^T x(1+\Theta[n])$. Depois extraímos a raiz, obtendo $fl(\|x\|_2) = \sqrt{fl(x^T x)}(1+\delta)$, onde $|\delta| < u$. Para facilitar as contas consideremos $(1+\delta)(1+\Theta[n])^{1/2}\|x\|_2 = (1+\Theta[n+1])\|x\|_2$, e portanto temos $\widehat{\phi} = \phi(1+\Theta[n+1])$.

Conseqüentemente,

$$\begin{aligned} \widehat{\beta} = fl\left(\frac{1}{x_1\widehat{\phi} + \widehat{\alpha}}\right) &= \frac{(1+\delta)^2}{(1+\delta')x_1\phi(1+\Theta[n+1]) + \alpha(1+\Theta[n])} \\ &= \frac{(1+\delta)^2}{(x_1\phi+\alpha)(1+\Theta[n+2])} = \beta(1+\Theta[2n+6]). \end{aligned}$$

Agora, nomeando $\omega = \widehat{\beta}\widehat{v}(\widehat{v}^T b)$, temos

$$fl(\omega) = fl(\widehat{\beta}\widehat{v}(\widehat{v}^T b)) = (1+\delta_1)(1+\delta_2)\widehat{\beta}\widehat{v}fl(\widehat{v}^T b), \qquad |\delta_1|, |\delta_2| \leq u.$$

De onde segue que

$$\begin{aligned} fl(\omega) &= (1+\Theta[2])(1+\Theta[2n+6])(1+\Theta[n+2])^2\beta v[v^T(b+\Delta b)] \\ &= (1+\Theta[4n+12])\beta vv^T(b+\Delta b), \end{aligned}$$

com $|\Delta b| \leq \Gamma(n)|b|$. Ou seja,

$$fl(\omega) = \beta v(v^T b) + \Delta\omega$$

e

$$\begin{aligned} |\Delta\omega| &= |\Theta[4n+12]\beta vv^T(b+\Delta b) + \beta vv^T\Delta b| \\ &\leq \Gamma(4n+12)|\beta vv^T b| + \Gamma(n)|\beta vv^T||b| + \Gamma(4n+12)\Gamma(n)|\beta vv^T||b| \\ &\leq \Gamma(5n+12)|\beta vv^T||b|. \end{aligned}$$

Conseqüentemente

$$\widehat{y} = fl(b - \widehat{\omega}) = b - \beta v(v^T b) - \Delta\omega + \Delta y_1, \qquad |\Delta y_1| \leq u|b - \widehat{\omega}|,$$

onde

$$\begin{aligned} |-\Delta\omega + \Delta y_1| &\leq |\Delta\omega| + |\Delta y_1| \leq \Gamma(5n+12)|\beta vv^T||b| + u|b - \widehat{\omega}| \\ &\leq \Gamma(5n+12)|\beta vv^T||b| + u|b| + u|\beta v(v^T b)| + u\Gamma(5n+12)|\beta vv^T||b| \\ &\leq \Gamma(5n+13)|\beta vv^T||b| + u|b|. \end{aligned}$$

Usando (2.19), o erro na norma 2 é dado por

$$\begin{aligned} \|-\Delta\omega + \Delta y_1\|_2 &\leq \Gamma(5n+13)\sqrt{\text{posto}(\beta vv^T)}\|\beta vv^T\|_2\|b\|_2 + u\|b\|_2 \\ &= \Gamma(5n+13)\|\beta vv^T\|_2\|b\|_2 + u\|b\|_2 = (\Gamma(10n+26)+u)\|b\|_2. \end{aligned}$$

Portanto $\widehat{y} = Pb + \Delta y$, onde $\|\Delta y\|_2 \leq \Gamma(10n+27)\|b\|_2$, lembrando que $\|\beta vv^T\|_2 = 2$. Agora, queremos descobrir ΔP tal que $\widehat{y} = (P + \Delta P)b$. De onde $\Delta P = \Delta y b^T / b^T b$ e, conseqüentemente, $\|\Delta P\|_F = \|\Delta P\|_2 = \|\Delta y\|_2 / \|b\|_2 \leq \Gamma(10n+27)$.

□

O Lema 2.8 mostra o erro direto (Δy) e o erro reverso (ΔP) da aplicação da transformação de Householder, obtida pelo Algoritmo 2.2, a um vetor $b \in \mathbb{R}^n$ qualquer.

Agora vamos analisar o erro direto e o erro reverso do processo da tridiagonalização.

Lema 2.9. *Considere a seqüencia de transformações*

$$A_{k+1} = Q_k A_k Q_k,$$

onde $A_1 = A \in \mathbb{R}^{n\times n}$ *e* Q_k *é uma matriz da forma (2.23). Então*

$$\widehat{A}_{k+1} = Q_k \cdots Q_1(A + \Delta A)Q_1 \cdots Q_k, \qquad \|\Delta A\|_F \leq \Gamma(20kn + 54k)\|A\|_F. \quad (2.25)$$

Demonstração. Seja $B_{k+1} = Q_k \ldots Q_1 A$. Então a j-ésima coluna de B_{k+1} é dada por $b_j^{k+1} = Q_k \ldots Q_1 a_j$. Pelo Lema 2.8 temos

$$\widehat{b}_j^{k+1} = (Q_k + \Delta Q_k) \cdots (Q_1 + \Delta Q_1)a_j,$$

onde cada ΔQ_i, para $i = 1 : k$, satisfaz $\|\Delta Q_i\|_p = \|\Delta P_i\|_p \leq \Gamma(10n+27)$, com $p = 2, F$. Pelo Lema 2.6, obtemos

$$\widehat{b}_j^{k+1} = Q_k \cdots Q_1(a_j + \Delta a_j),$$

com

$$\begin{aligned}\|\Delta a_j\|_2 &= \left\| \prod_{i=1}^{k} Q_{k+1-i} \left(\prod_{i=1}^{k} (Q_i + \Delta Q_i) a_j - \prod_{i=1}^{k} Q_i a_j \right) \right\|_2 \\ &\leq \left\| \prod_{i=1}^{k} (Q_i + \Delta Q_i) - \prod_{i=1}^{k} Q_i \right\|_2 \|a_j\|_2 \\ &\leq \left((1 + \Gamma(10n + 27))^k - 1 \right) \prod_{i=1}^{k} \|Q_i\|_2 \|a_j\|_2.\end{aligned}$$

Dado que $\|Q_i\|_2 = \|P_i\|_2 = 1$ para todo i, então

$$\|\Delta a_j\|_2 \leq \left((1 + \Gamma(10n + 27))^k - 1 \right) \|a_j\|_2.$$

Pelo Lema 2.5, $(1 + \Gamma(10n + 27))^k - 1 \leq \Gamma(10kn + 27k)$. Portanto,

$$\|\Delta a_j\|_2 \leq \Gamma(10kn + 27k) \|a_j\|_2.$$

Conseqüentemente, $\widehat{B}_{k+1} = Q_k \cdots Q_1 (A + \Delta \tilde{A})$, onde

$$\|\Delta \tilde{A}\|_F \leq \Gamma(10kn + 27k) \|A\|_F.$$

Voltando a A_{k+1}, temos $A_{k+1} = B_{k+1} Q_1 \cdots Q_k$ e

$$\widehat{A}_{k+1} = (\widehat{B}_{k+1} + \Delta B_{k+1}) Q_1 \cdots Q_k.$$

Se $(\Delta b_j^{k+1})^T$ é j-ésima linha de ΔB_{k+1} e $(\widehat{b}_j^{k+1})^T$ é j-ésima linha da matriz $\widehat{B}_{k+1}$ então

$$\|(\Delta b_j^{k+1})^T\|_2 \leq \Gamma(10kn + 27k) \|(\widehat{b}_j^{k+1})^T\|_2,$$

de onde

$$\|\Delta B_{k+1}\|_F = \Gamma(10kn + 27k) \|\widehat{B}_{k+1}\|_F.$$

Agora

$$\begin{aligned}\widehat{A}_{k+1} &= \left(Q_k \cdots Q_1 (A + \Delta \tilde{A}) + \Delta B_{k+1} \right) Q_1 \cdots Q_k \\ &= Q_k \cdots Q_1 \left(A + \Delta \tilde{A} + (Q_1 \cdots Q_k) \Delta B_{k+1} \right) Q_1 \cdots Q_k,\end{aligned}$$

ou seja, $\Delta A = \Delta \tilde{A} + (Q_1 \cdots Q_k) \Delta B_{k+1}$. Mas

$$\|\Delta A\|_F \leq \|\Delta \tilde{A}\|_F + \prod_{i=1}^{k} \|Q\|_2 \|\Delta B_{k+1}\|_F = \|\Delta \tilde{A}\|_F + \|\Delta B_{k+1}\|_F,$$

de onde, lembrando que $\widehat{B}_{k+1} = Q_k \cdots Q_1(A + \Delta\tilde{A})$,

$$\begin{aligned}\|\Delta A\|_F &\leq \|\Delta\tilde{A}\|_F + \Gamma(10kn + 27k)\|\widehat{B}_{k+1}\|_F \\ &\leq \|\Delta\tilde{A}\|_F + \Gamma(10kn + 27k)\left(\|A\|_F + \|\Delta\tilde{A}\|_F\right) \\ &\leq \left(2\Gamma(10kn + 27k) + (\Gamma(10kn + 27k))^2\right)\|A\|_F \\ &\leq \Gamma(20kn + 54k)\|A\|_F.\end{aligned}$$

□

Lema 2.10. *Seja A_{k+1} definida como no lema anterior, então*

$$\|\widehat{A}_{k+1} - A_{k+1}\|_F \leq \Gamma(20kn + 54k)\|A\|_F. \tag{2.26}$$

Demonstração. Como no Lema anterior seja $B_{k+1} = Q_k \cdots Q_1 A$. Então

$$\widehat{b}_j^{k+1} = (Q_k + \Delta Q_k) \cdots (Q_1 + \Delta Q_1)a_j,$$

onde cada ΔQ_i, para $i = 1 : k$, satisfaz $\|\Delta Q_i\|_p = \|\Delta P_i\|_p \leq \Gamma(10n + 27)$, com $p = 2$, F. Dado que $\|Q_i\|_2 = \|P_i\|_2 = 1$ para todo i, então, usando Lema 2.6,

$$\begin{aligned}\|\widehat{b}_j^{k+1} - b_j^{k+1}\|_2 &\leq \left\|\prod_{i=1}^{k}(Q_i + \Delta Q_i) - \prod_{i=1}^{k} Q_i\right\|_2 \|a_j\|_2 \\ &\leq \Gamma(10kn + 27k)\|a_j\|_2.\end{aligned}$$

De onde

$$\|\widehat{B}_{k+1} - B_{k+1}\|_F \leq \Gamma(10kn + 27k)\|A\|_F.$$

Voltando a A_{k+1}, temos $A_{k+1} = B_{k+1}Q_1 \cdots Q_k$ e

$$\widehat{A}_{k+1} = \widehat{B}_{k+1}(Q_1 + \Delta Q_1) \cdots (Q_k + \Delta Q_k).$$

Portanto

$$\begin{aligned}\|\widehat{A}_{k+1} - A_{k+1}\| \leq &\|\widehat{B}_{k+1}\|_F \left\|\prod_{i=1}^{k}(Q_{k-i+1} + \Delta Q_{k-i+1}) - \prod_{i=1}^{k} Q_{k-i+1}\right\|_F \\ &+ \|\widehat{B}_{k+1} - B_{k+1}\|_F \prod_{i=1}^{k}\|Q_i\|_F \\ \leq &\Gamma(10kn + 27k)\|\widehat{B}_{k+1}\|_F + \Gamma(10kn + 27k)\|A\|_F.\end{aligned}$$

Usando expressão $\widehat{B}_{k+1} = Q_k \cdots Q_1(A + \Delta\tilde{A})$ obtida no Lema anterior, onde

$$\|\Delta\tilde{A}\|_F \leq \Gamma(10kn + 27k)\|A\|_F,$$

temos

$$\begin{aligned}\|\widehat{A}_{k+1} - A_{k+1}\| &\leq \Gamma(10kn + 27k)\left(\|A\|_F + \|\Delta\tilde{A}\|_F\right) + \Gamma(10kn + 27k)\|A\|_F \\ &\leq \Gamma(20kn + 54k)\|A\|_F.\end{aligned}$$

□

Observamos que o erro reverso e o erro direto fornecidos pelos Lemas 2.9 e 2.10, respectivamente, possuem a mesma majoração, e no $(n-2)$-ésimo passo, quando A_{n-1} se torna matriz tridiagonal, são dados por

$$\begin{aligned}\|\widehat{A}_{n-1} - A_{n-1}\|_F &\leq \Gamma(20n^2 + 14n - 54)\|A\|_F, \\ \|\Delta A\|_F &\leq \Gamma(20n^2 + 14n - 54)\|A\|_F.\end{aligned} \tag{2.27}$$

Capítulo 3

Análise de erro da transformação de Householder generalizada

Nesse capítulo faremos a análise de erro direto e reverso da transformação de Householder generalizada ótima e do algoritmo de tridiagonalização usando essa transformação.

Na primeira sessão definimos a transformação de Householder generalizada e o algoritmo da transformação de Householder generalizada ótima introduzido por Golub, Yuan, Biloti e Ramos em [4]. Na segunda seção expomos o algoritmo de tridiagonalização. E na terceira realizaremos a análise de erro direto e reverso.

3.1 Transformação de Householder generalizada

Em seu artigo [10] publicado em 1963, LaBudde estabeleceu um algoritmo para reduzir uma matriz arbitrária a uma matriz tridiagonal semelhante através da transformação definida por

$$P = I + awv^T, \tag{3.1}$$

cuja inversa é dada por

$$P^{-1} = I + bwv^T, \tag{3.2}$$

onde

$$\frac{a+b}{ab} = -w^T v. \tag{3.3}$$

Tomando $a = b = -2/w^T w$ e $v = w$ pode ser facilmente visto que a transformação de Householder generalizada se reduz a transformação de Householder.

Um estudo detalhado sobre transformação de Householder generalizada e suas propriedades foi feito por Golub, Yuan, Biloti e Ramos em 2002 [4]. A seguir expomos um breve resumo desse trabalho omitindo as demonstrações.

Teorema 3.1. *Seja P transformação de Householder generalizada com v e w linearmente independentes. Então os valores singulares de P são σ_1, 1 e σ_2, satisfazendo $\sigma_1 \geq 1 \geq \sigma_2 > 1$ para $a \neq -\frac{1}{v^T w}$, onde*

$$\sigma_1 = \sqrt{1 + aw^T v + \frac{1}{2}a^2\|v\|_2^2\|w\|_2^2 + \sqrt{a^2\|v\|_2^2\|w\|_2^2(1 + aw^T v + \frac{1}{4}a^2\|v\|_2^2\|w\|_2^2)}}$$

e

$$\sigma_1 = \sqrt{1 + aw^T v + \frac{1}{2}a^2\|v\|_2^2\|w\|_2^2 - \sqrt{a^2\|v\|_2^2\|w\|_2^2(1 + aw^T v + \frac{1}{4}a^2\|v\|_2^2\|w\|_2^2)}}$$

Esse teorema garante que a norma 2 da transformação de Householder generalizada é sempre maior ou igual a 1.

Teorema 3.2. *Seja $v^T w \neq 0$, então o número de condição na norma 2 da transformação de Householder generalizada P atinge o seu mínimo quando $a = -\frac{2}{w^T v}$.*

Note que quando $v^T w \neq 0$ temos $a = b = -\frac{2}{w^T v}$, pois $-w^T u = \frac{a+b}{ab}$.

Agora suponhamos que a matriz dada A tem a seguinte forma

$$A = \begin{pmatrix} \tau & y^T \\ x & A_{n-1} \end{pmatrix}.$$

Para transformar A em uma matriz tridiagonal semelhante, a transformação de Householder generalizada deve satisfazer as seguintes condições:

$$(I + avw^T)x = k_1 e_1,$$

e

$$(I + bwv^T)y = k_2 e_1.$$

Pelo Teorema 3.2, a opção ótima de a e b, em relação ao número de condição na norma 2, é $a = b = -\frac{2}{w^T v}$. Denotamos a transformação de Householder generalizada com opção ótima de a como a transformação de Householder generalizada ótima e temos o seguinte resultado.

Teorema 3.3. *Dados os vetores x e y tais que $x_1y_1x^Ty > 0$, então a transformação de Householder generalizada ótima é dada por*

$$\begin{cases} v = x - k_1e_1, \\ w = y - k_2e_1, \\ k_1 = -\,sign(y_1)\,sign(x^Ty)\sqrt{\frac{x_1}{y_1}x^Ty}, \\ k_2 = -\,sign(x_1)\,sign(x^Ty)\sqrt{\frac{y_1}{x_1}x^Ty}, \\ a = b = -\dfrac{1}{x^Ty + sign(x^Ty)\sqrt{x_1y_1x^Ty}}. \end{cases}$$

Observação 3.1. *Pelo Teorema 3.3, $v^Tw \neq 0$ implica em opção ótima $a = b = -\frac{2}{w^Tv}$ que leva a $x_1y_1x^Ty > 0$.*

Utilizando a observação acima obtemos o seguinte lema.

Lema 3.4. *Dados x e y, $v^Tw = 0$ sempre que $x_1y_1x^Ty < 0$.*

Teorema 3.5. *Dados os vetores x e y tais que $x_1y_1x^Ty < 0$, então a transformação de Householder generalizada ótima é dada por*

$$\begin{cases} v = x - k_1e_1, \\ w = y - k_2e_1, \\ k_1 = sign(y_1)\,sign(x^Ty)\dfrac{|x^Ty| + \sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}}{|y_1|}, \\ k_2 = -\,sign(x_1)\,sign(x^Ty)\dfrac{|y_1(x^Ty)|}{|x^Ty| + \sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}}, \\ a = -b = \dfrac{sign(x^Ty)}{\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}}. \end{cases}$$

Dos Teoremas 3.3 e 3.5 segue o algoritmo da transformação de Householder generalizada ótima.

Algoritmo 3.1. **Transformação de Householder generalizada ótima e sua Inversa**

1. *Dados x e y tais que $(x^Ty)x_1y_1 \neq 0$;*

2. *Se $(x^Ty)x_1y_1 > 0$*
   então
   $k_1 = -\,sign(y_1)\,sign(x^Ty)\sqrt{\frac{x_1}{y_1}x^Ty}$,

$k_2 = -sign(x_1)\, sign(x^T y)\sqrt{\frac{y_1}{x_1} x^T y},$

$a = b = -\left(x^T y + sign(x^T y)\sqrt{x_1 y_1 x^T y}\right)^{-1};$

senão

$k_1 = sign(y_1)\, sign(x^T y)\frac{|x^T y| + \sqrt{|x^T y|^2 + |x_1 y_1||x^T y|}}{|y_1|},$

$k_2 = -sign(x_1)\, sign(x^T y)\frac{|y_1 (x^T y)|}{|x^T y| + \sqrt{|x^T y|^2 + |x_1 y_1||x^T y|}},$

$a = -b = sign(x^T y)\left(\sqrt{|x^T y|^2 + |x_1 y_1||x^T y|}\right)^{-1};$

3. $w = x - k_1 e_1$ *e* $v = y - k_2 e_1$;

4. $P = I + awv^T$ *e* $P^{-1} = I + bwv^T$;

3.2 Tridiagonalização

No capítulo anterior (Seção 2.5.1) mostramos como pode ser feita a tridiagonalização de uma matriz simétrica aplicando as transformações de Householder. No caso das transformações de Householder generalizadas podemos tridiagonalizar uma matriz qualquer, aplicando procedimento similar.

Seja $A \in \mathbb{R}^{n\times n}$. Definimos a seguinte seqüencia de matrizes

$$A_{k+1} = Q_k A_k Q_k^{-1}, \tag{3.4}$$

onde $A_1 = A$,

$$Q_k = \begin{pmatrix} I_k & 0 \\ 0 & P_k \end{pmatrix}, \qquad Q_k^{-1} = \begin{pmatrix} I_k & 0 \\ 0 & P_k^{-1} \end{pmatrix} \tag{3.5}$$

e P_k, $P_k^{-1} \in \mathbb{R}^{(n-k)\times(n-k)}$ são matrizes de Householder generalizadas ótimas definidas na seção anterior. LaBudde [10] mostrou que após $n-2$ passos obtemos uma matriz tridiagonal semelhante à matriz A.

De fato, suponhamos

$$A_k = \left(\begin{array}{ccc|c} & & & 0 \\ & T_k & & \vdots \\ & & & y^T \\ \hline 0 & \cdots & x & A_{[n-k]} \end{array}\right), \tag{3.6}$$

onde $A_{[n-k]} = A_k(k+1:n, k+1:n)$. Sejam P_k e P_k^{-1} tais que $P_k x = k_1 e_1$ e $P_k^{-1} y = k_2 e_1$, então

$$A_{k+1} = Q_k A_k Q_k^{-1} = \left(\begin{array}{c|c} & 0 \\ T_k & \vdots \\ & k_2 e_1^T \\ \hline 0 \quad \cdots \quad k_1 e_1 & P_k A_{[n-k]} P_k^{-1} \end{array}\right) \tag{3.7}$$

e de (3.1) e (3.2) segue que

$$P_k A_{[n-k]} P_k^{-1} = A_{[n-k]} + awv^T A_{[n-k]} + bA_{[n-k]} wv^T + abwv^T A_{[n-k]} wv^T. \tag{3.8}$$

Logo após $n-2$ passos a matriz A_{n-1} é uma matriz tridiagonal.

Pelo fato da transformação de Householder generalizada não funcionar quando $x^T y = 0$ ou $x_1 y_1 = 0$, (ver Algoritmo 3.1) é introduzida uma correção do *Break-Down* [4], para evitar essa situação.

Algoritmo 3.2. **Tridiagonalização pela transformação de Householder generalizada ótima**

1. *Para* $i = 1 : n-2$

   1.1. $x = A(i+1:n, i)$, $y = A(i, i+1:n)$;

   1.2. *Seja* $\mathcal{J} \subset \{i+1, \ldots, n\}$ *o conjunto de índices para quais é satisfeita a condição* $x^T y \neq 0$, *trocando as colunas* i *e* $j \in \{i+1, \ldots, n\}$, *e linhas* i *e* j;

   1.3. *Se* $\mathcal{J}$ *é vazio, então chama Algoritmo 3.3 e redefina* $\mathcal{J}$ *como em 1.2;*

   1.4. *Seja* J *seja o índice tal que*

   $$|A(J,i) \cdot A(i,J)| = max_{j \in \mathcal{J}} \{|A(j,i) \cdot A(i,j)|\};$$

   1.5. *Trocar colunas* $i+1$ *e* J, *e linhas* $i+1$ *e* J;

   1.6. *Chamar Algoritmo 3.1;*

   1.7. $t = A(i+1:n, i+1:n)^T v$,
   $p = A(i+1:n, i+1:n) w$,
   $\phi = v^T p$;

1.8. Para k=i+1:n

$\mu = w_k a,$

$\nu = (p_k + \mu\phi)b;$

Para j=i+1:n

$A(k,j) = A(k,j) + \mu t_j + \nu v_j;$

1.9. $A(i+1,i) = k_1,$

$A(i,i+1) = k_2,$

$A(i,i+2:n) = A(i+2:n,i) = 0.$

Algoritmo 3.3. **Correção do** *Break-Down*

1. *Achar* j *tal que* $s = r^T w \neq 0$, $r_1 \neq 0$ *e* $w_1 \neq 0$, *onde* $r = A(i+1:n,j)$ *e* $w = A(j,i+1:n)$;

2. *Escolher* α *grande suficiente de tal forma que* $\alpha s + x^T w - r^T y \neq 0$, $x_1 + \alpha r_1 \neq 0$ *e* $y_1 - \alpha w_1 \neq 0$;

3. $A(i:n,i+1) = A(i:n,i+1) + \alpha A(i:n,j),$
$A(i+1,i:n) = A(i+1,i:n) - \alpha A(j,i:n);$

4. $x = A(i+1:n,i)$, $y = A(i,i+1:n)$.

3.3 Alguns resultados teóricos

Expomos a seguir alguns resultados envolvendo produto interno que serão usados nos cálculos posteriores.

Observação: Consideraremos $zu \leq 1$ para qualquer coeficiente de erro z encontrado a partir dessa seção.

Lema 3.6. *Sejam* x, y *vetores de* $\mathbb{R}^n$ *e* $x^T y > 0$ *o produto interno obtido pelo Algoritmo 2.1. Se* $s = \sqrt{x^T y}$ *então*

$$\widehat{s} = fl(s) = s(1 + \Theta[n+1]\frac{|x|^T|y|}{x^T y}). \tag{3.9}$$

Demonstração. Usando (2.5) podemos escrever

$$fl(\sqrt{x^T y}) = (1+\delta)\sqrt{fl(x^T y)}, \qquad |\delta| \leq u.$$

De (2.16) obtemos

$$\sqrt{fl(x^Ty)} = \sqrt{x^Ty\left(1+\Theta[n]\frac{|x|^T|y|}{x^Ty}\right)}.$$

Conseqüentemente

$$\begin{aligned}\frac{|\widehat{s}-s|}{|s|} &= \frac{|fl(\sqrt{x^Ty})-\sqrt{x^Ty}|}{\sqrt{x^Ty}} = \left|(1+\delta)\sqrt{1+\Theta[n]\frac{|x|^T|y|}{x^Ty}}-1\right| \\ &\leq (1+u)\left(1+\Gamma(n)\frac{|x|^T|y|}{x^Ty}\right)-1 \leq \frac{|x|^T|y|}{x^Ty}(\Gamma(n)+u+u\Gamma(n)) \\ &\leq \Gamma(n+1)\frac{|x|^T|y|}{x^Ty}.\end{aligned}$$

□

Lema 3.7. *Sejam x, y vetores de $\mathbb{R}^n$ e $x^Ty > 0$ calculado pelo Algoritmo 2.1 Se $t = \frac{1}{x^Ty}$, então*

$$\widehat{t} = fl(t) = t(1+\Theta[(\alpha+1)(n+1)]), \tag{3.10}$$

onde $\alpha = \frac{|x|^T|y|}{|x^Ty|}$.

Demonstração. Aplicando modelo (2.6), $\widehat{t}$ pode ser escrito como

$$\widehat{t} = fl\left(\frac{1}{x^Ty}\right) = \frac{1}{fl(x^Ty)}\frac{1}{(1+\delta)}, \qquad |\delta| \leq u.$$

Interpretando $fl(x^Ty)$ pela equação (2.13) obtemos

$$\widehat{t} = \frac{1}{x^T(y+\Delta y)}, \qquad |\Delta y| \leq \Gamma(n+1)|y|.$$

Logo,

$$\begin{aligned}\frac{|\widehat{t}-t|}{|t|} &= \frac{\left|\frac{1}{x^T(y+\Delta y)}-\frac{1}{x^Ty}\right|}{\left|\frac{1}{x^Ty}\right|} = |x^Ty|\left|\frac{1}{x^Ty(1+\frac{x^T\Delta y}{x^Ty})}-\frac{1}{x^Ty}\right| \\ &= \left|\frac{-\frac{x^T\Delta y}{x^Ty}}{1+\frac{x^T\Delta y}{x^Ty}}\right|\end{aligned}$$

Lembrando que, por hipótese, $x^Ty > 0$, temos

$$\left|\frac{-\frac{x^T\Delta y}{x^Ty}}{1+\frac{x^T\Delta y}{x^Ty}}\right| \leq \frac{\Gamma(n+1)\frac{|x|^T|y|}{x^Ty}}{1-\Gamma(n+1)\frac{|x|^T|y|}{x^Ty}} = \frac{\alpha\Gamma(n+1)}{1-\alpha\Gamma(n+1)}.$$

Agora,

$$\frac{\alpha\Gamma(n+1)}{1-\alpha\Gamma(n+1)} = \frac{\frac{\alpha(n+1)u}{1-(n+1)u}}{1-\frac{\alpha(n+1)u}{1-(n+1)u}} = \frac{\alpha(n+1)u}{1-(\alpha+1)(n+1)u} \leq \frac{(\alpha+1)(n+1)u}{1-(\alpha+1)(n+1)u}$$
$$= \Gamma((\alpha+1)(n+1)),$$

ou seja,

$$\frac{|\widehat{t}-t|}{|t|} \leq \Gamma((\alpha+1)(n+1)).$$

□

Note que o Lema 2.5 também é válido para índices reais, portanto usando a relação (iv) no Lema 3.6 podemos escrever a equação (3.9) na seguinte forma

$$fl(\sqrt{x^Ty}) = \sqrt{x^Ty}(1+\Theta[\alpha n+1]), \tag{3.11}$$

onde $\alpha = \frac{|x|^T|y|}{|x^Ty|}$.

3.4 Análise de erro da transformação de Householder generalizada

Nessa seção faremos a análise de erro da transformação de Householder generalizada ótima. As demonstrações por extenso serão feitas apenas para P, pois os resultados obtidos se aplicam também a P^{-1}. De fato, no caso quando $(x^Ty)x_1y_1 > 0$, P e P^{-1} são simplesmente iguais, e quando $(x^Ty)x_1y_1 < 0$ a diferença está apenas no sinal dos coeficientes a e b ($a = -b$) o que não afeta a análise.

Primeiramente, analisaremos o erro quando $(x^Ty)x_1y_1 > 0$.

Lema 3.8. *Sejam x, $y \in \mathbb{R}^n$ vetores tais que $(x^Ty)x_1y_1 > 0$. Então os coeficientes $\widehat{k}_1$, $\widehat{k}_2$, $\widehat{a}$ e $\widehat{b}$ e os vetores $\widehat{w}$ e $\widehat{v}$, obtidos pelo Algoritmo 3.1, são dados por*

$$\widehat{k}_1 = k_1(1+\Theta[\alpha n+3]), \tag{3.12}$$
$$\widehat{k}_2 = k_2(1+\Theta[\alpha n+3]), \tag{3.13}$$
$$\widehat{a} = \widehat{b} = a(1+\Theta[(\alpha+1)(n+5)]), \tag{3.14}$$
$$\widehat{w} = w(1+\Theta[\alpha n+4]), \tag{3.15}$$
$$\widehat{v} = v(1+\Theta[\alpha n+4]), \tag{3.16}$$

onde $\alpha = \frac{|x|^T|y|}{|x^Ty|}$.

Demonstração. As relações (3.12) e (3.13) são imediatas, basta notar que

$$fl\left(\sqrt{\frac{x_1}{y_1}x^Ty}\right) = (1+\delta)\sqrt{\frac{x_1}{y_1}x^T(y+\Delta y)}, \qquad |\delta| \leq u, |\Delta y| \leq \Gamma(n+2).$$

e aplicar equação 3.11.

Para mostrar relação (3.14), em primeiro lugar, vamos escrever $\widehat{a}$ na seguinte forma:

$$\begin{aligned}\widehat{a} &= -fl\left(\left(x^Ty + \text{sign}(x^Ty)\sqrt{x_1y_1x^Ty}\right)^{-1}\right)\\ &= -\left((1+\delta_1)(1+\delta_2)\left[fl(x^Ty) + \text{sign}(x^Ty)fl\left(\sqrt{x_1y_1x^Ty}\right)\right]\right)^{-1},\end{aligned}$$

onde $|\delta_1|, |\delta_2| \leq u$. Usando equações (2.13), (2.16) e Lema 3.6 temos

$$(1+\delta_1)(1+\delta_2)fl(x^Ty) = x^Ty\left(1+\Theta[n+2]\frac{|x|^T|y|}{|x^Ty|}\right),$$

e

$$(1+\delta_1)(1+\delta_2)fl(\sqrt{x_1y_1x^Ty}) = \sqrt{x_1y_1x^Ty}\left(1+\Theta[n+5]\frac{|x|^T|y|}{|x^Ty|}\right).$$

Assim como (x^Ty) e $(\text{sign}(x^Ty)\sqrt{x_1y_1x^Ty})$ possuem o mesmo sinal, temos

$$\widehat{a} = -\left(\left(1+\Theta[n+5]\frac{|x|^T|y|}{|x^Ty|}\right)\left[x^Ty + \text{sign}(x^Ty)\sqrt{x_1y_1x^Ty}\right]\right)^{-1}.$$

Repetindo os passos da demonstração do Lema 3.7 obtemos

$$\widehat{a} = a(1+\Theta[(\alpha+1)(n+5)]).$$

Por fim, como $\text{sign}(x_1) = -\text{sign}(k_1)$, tem-se

$$\begin{aligned}\widehat{w}_1 &= (x_1 - \widehat{k}_1)(1+\delta) = (x_1 - k_1)(1+\Theta[\alpha n+3])(1+\delta)\\ &= w_1(1+\Theta[\alpha n+4]), \qquad |\delta| \leq u.\end{aligned}$$

O mesmo vale para v_1. □

O próximo resultado descreve o erro relacionado à aplicação da transformação de Householder generalizada a um vetor.

Lema 3.9. *Sejam $z \in \mathbb{R}^n$ e $y = Pz = (I + awv^T)z = z + au(v^Tz)$. Então*

$$\begin{aligned}\widehat{y} &= fl(z + \widehat{a}\widehat{w}(\widehat{v}^Tz)) = (P+\Delta P)z,\\ &\|\Delta P\|_p \leq \Gamma((6\alpha+4)n + 22\alpha + 19)\|P\|_2, \quad p = 2, F. \qquad (3.17)\end{aligned}$$

onde $\widehat{a}$, $\widehat{w}$ e $\widehat{v}$ são dados pelo Lema anterior.

Demonstração. A demonstração é análoga a que foi feita na segunda parte do Lema 2.8. Definindo $\omega = \widehat{a}\widehat{w}(\widehat{v}^T z)$ e usando Lema 2.5 temos

$$\begin{aligned}\widehat{\omega} &= (1+\delta_1)(1+\delta_2)(1+\Theta[(\alpha+1)(n+5)])(1+\Theta[\alpha n+4])^2 au(v^T(z+\Delta z)) \\ &= (1+\Theta[(3\alpha+1)n+5\alpha+13])au(v^T(z+\Delta z)), \qquad |\Delta z| \leq \Gamma(n)|z|.\end{aligned}$$

Ou seja,

$$\widehat{\omega} = awv^T z + \Delta\omega,$$

onde

$$\begin{aligned}|\Delta\omega| =& |awv^T\Delta z + \Theta[(3\alpha+1)n+5\alpha+13]awv^T(z+\Delta z)| \\ \leq& \Gamma(n)|awv^T||z| + \Gamma((3\alpha+1)n+5\alpha+13)|awv^T z| \\ &+ \Gamma(n)\Gamma((3\alpha+1)n+5\alpha+13)|awv^T||z| \\ \leq& \Gamma((3\alpha+2)n+5\alpha+13)|awv^T||z|.\end{aligned}$$

Usando (2.19) e o fato de que $\text{posto}(awv^T) = 1$, o erro na norma 2 é dado por

$$\begin{aligned}\|\Delta\omega\|_2 &\leq \Gamma((3\alpha+2)n+5\alpha+13)\|awv^T\|_2\|z\|_2 \\ &\leq \Gamma((3\alpha+2)n+5\alpha+13)(\|P\|_2+1)\|z\|_2.\end{aligned}$$

Pelo Teorema 3.1 o maior valor singular σ_1 da transformação de Householder generalizada é maior ou igual a 1, quando $a \neq -\frac{1}{w^T v}$. Assim como a transformação de Householder generalizada ótima satisfaz essa hipótese, $\|P\|_2 = \sigma_1 \geq 1$. Portanto

$$\begin{aligned}\|\Delta\omega\|_2 &\leq 2\Gamma((3\alpha+2)n+5\alpha+13)\|P\|_2\|z\|_2 \\ &\leq \Gamma((6\alpha+4)n+10\alpha+26)\|P\|_2\|z\|_2.\end{aligned}$$

Voltando à expressão de y temos

$$\widehat{y} = fl(z+\widehat{\omega}) = (Pz-\Delta\omega)(1+\delta) = y + \Delta y$$

onde

$$\begin{aligned}\|\Delta y\|_2 \leq& \|\delta Pz - \Delta\omega + \delta\Delta\omega\|_2 \\ \leq& u\|Pz\|_2 + \Gamma((6\alpha+4)n+10\alpha+26)\|P\|_2\|z\|_2 \\ &+ u\Gamma((6\alpha+4)n+10\alpha+26)\|P\|_2\|z\|_2 \\ \leq& (u+\Gamma((6\alpha+4)n+10\alpha+26)+u\Gamma((6\alpha+4)n+10\alpha+26))\,\|P\|_2\|z\|_2 \\ \leq& \Gamma((6\alpha+4)n+10\alpha+27)\|P\|_2\|z\|_2.\end{aligned}$$

Agora, queremos achar ΔP tal que $\widehat{y} = (P + \Delta P)z$. Então tomando $\Delta P = \Delta y z^T / z^T z$ tem-se $\|\Delta P\|_F = \|\Delta P\|_2 = \|\Delta y\|_2 / \|z\|_2 \leq \Gamma((6\alpha + 4)n + 10\alpha + 27)\|P\|_2$ □

Assim como $P = P^{-1}$, o resultado do Lema se aplica também a $y = (P^{-1})^T z$, ou seja,

$$\widehat{y} = (P^{-1} + \Delta P^{-1})^T z, \quad \|\Delta P^{-1}\|_p \leq \Gamma((6\alpha + 4)n + 10\alpha + 27)\|P^{-1}\|_2, \tag{3.18}$$

onde $p = 2, F$.

O próximo passo é analisar o erro no processo de tridiagonalização.

Lema 3.10. *Considere a seqüencia de transformações*

$$A_{k+1} = Q_k A_k Q_k^{-1}, \tag{3.19}$$

onde $A_1 = A \in \mathbb{R}^{n \times n}$,

$$Q_k = \begin{pmatrix} I_k & 0 \\ 0 & P_k \end{pmatrix}, \qquad Q_k^{-1} = \begin{pmatrix} I_k & 0 \\ 0 & P_k^{-1} \end{pmatrix} \tag{3.20}$$

e $P_k, P_k^{-1} \in \mathbb{R}^{(n-k)\times(n-k)}$ *são transformações de Householder generalizadas ótimas com* $(x^T y)x_1 y_1 > 0$. *Então*

$$\widehat{A}_{k+1} = Q_k \cdots Q_1 (A + \Delta A) Q_1^{-1} \cdots Q_k^{-1}, \tag{3.21}$$

onde

$$\|\Delta A\|_F \leq \Gamma \left(\left(12 \sum_{i=1}^{k} \alpha_i + 8k \right) n + 20 \sum_{i=1}^{k} \alpha_i + 54k \right) \|A\|_F \prod_{i=1}^{k} \kappa_2(P_i)^3$$

e $\kappa_2(P_i)$ *é número de condição de* P_i.

Demonstração. Seja $B_{k+1} = Q_k \ldots Q_1 A$. Então j-ésima coluna de B_{k+1} é dada por $b_j^{k+1} = Q_k \ldots Q_1 a_j$. Pelo Lema 3.9 temos

$$\widehat{b}_j^{k+1} = (Q_k + \Delta Q_k) \cdots (Q_1 + \Delta Q_1) a_j,$$

onde cada ΔQ_i, para $i = 1 : k$, satisfaz $\|\Delta Q_i\|_p = \|\Delta P_i\|_p \leq \Gamma((6\alpha_i + 4)n + 10\alpha_i + 27)\|P_i\|_2$, com $p = 2, F$. Dado que $\|P_i\|_2 \geq 1$, tem-se $\|Q_i\|_2 = \|P_i\|_2$ para todo i e usando Lema 2.6 obtemos

$$\widehat{b}_j^{k+1} = Q_k \cdots Q_1 (a_j + \Delta \tilde{a}_j),$$

com

$$\begin{aligned}\|\Delta\tilde{a}_j\|_2 &= \left\|\prod_{i=1}^{k} Q_{k+1-i}^{-1}\left(\prod_{i=1}^{k}(Q_i+\Delta Q_i)a_j - \prod_{i=1}^{k} Q_i a_j\right)\right\|_2 \\ &\leq \prod_{i=1}^{k}\|Q_i^{-1}\|_2 \left\|\prod_{i=1}^{k}(Q_i+\Delta Q_i) - \prod_{i=1}^{k} Q_i\right\|_2 \|a_j\|_2 \\ &\leq \left(\prod_{i=1}^{k}(1+\Gamma((6\alpha_i+4)n+10\alpha_i+27)) - 1\right)\prod_{i=1}^{k}\|Q_i^{-1}\|_2 \prod_{i=1}^{k}\|Q_i\|_2\|a_j\|_2.\end{aligned}$$

Aplicando o Lema 2.5 a $\prod_{i=1}^{k}(1+\Gamma((6\alpha_i+4)n+10\alpha_i+27))$ temos

$$\begin{aligned}\prod_{i=1}^{k}(1+\Gamma((6\alpha_i+4)n+10\alpha_i+27)) &\leq 1+\Gamma\left(\sum_{i=1}^{k}[(6\alpha_i+4)n+10\alpha_i+27]\right) \\ &= 1+\Gamma\left(\left(6\sum_{i=1}^{k}\alpha_i+4k\right)n+10\sum_{i=1}^{k}\alpha_i+27k\right).\end{aligned}$$

Para facilitar a leitura consideremos

$$\chi = \left(6\sum_{i=1}^{k}\alpha_i+4k\right)n+20\sum_{i=1}^{k}\alpha_i+27k.$$

Então,

$$\begin{aligned}\|\Delta\tilde{a}_j\|_2 &\leq \Gamma(\chi)\prod_{i=1}^{k}\|P_i^{-1}\|_2\|P_i\|_2\|a_j\|_2 \\ &= \Gamma(\chi)\prod_{i=1}^{k}\kappa_2(P_i)\|a_j\|_2,\end{aligned}$$

onde, $\kappa_2(P_i)$ é número de condição de P_i. Conseqüentemente $\widehat{B}_{k+1} = Q_k\cdots Q_1(A+\Delta\tilde{A})$, onde

$$\|\Delta\tilde{A}\|_F \leq \Gamma(\chi)\|A\|_F\prod_{i=1}^{k}\kappa_2(P_i).$$

Voltando a A_{k+1}, temos que $A_{k+1} = B_{k+1}Q_1^{-1}\cdots Q_k^{-1}$ e

$$\widehat{A}_{k+1} = (\widehat{B}_{k+1}+\Delta B_{k+1})Q_1^{-1}\cdots Q_k^{-1}.$$

Se $(\Delta b_j^{k+1})^T$ é j-ésima linha da matriz ΔB_{k+1} e $(\widehat{b}_j^{k+1})^T$ é j-ésima linha da matriz $\widehat{B}_{k+1}$, então

$$\|(\Delta b_j^{k+1})^T\|_2 \leq \Gamma(\chi)\prod_{i=1}^{k}\kappa_2(P_i)\|(\widehat{b}_j^{k+1})^T\|_2,$$

de onde

$$\|\Delta B_{k+1}\|_F \leq \Gamma(\chi)\|\widehat{B}_{k+1}\|_F \prod_{i=1}^{k} \kappa_2(P_i).$$

Agora

$$\begin{aligned}\widehat{A}_{k+1} &= \left(Q_k \cdots Q_1(A + \Delta\tilde{A}) + \Delta B_{k+1}\right) Q_1^{-1} \cdots Q_k^{-1} \\ &= Q_k \cdots Q_1 \left(A + \Delta\tilde{A} + (Q_1^{-1} \cdots Q_k^{-1})\Delta B_{k+1}\right) Q_1^{-1} \cdots Q_k^{-1},\end{aligned}$$

ou seja, $\Delta A = \Delta\tilde{A} + (Q_1^{-1} \ldots Q_k^{-1})\Delta B_{k+1}$. Utilizando o fato de que

$$\|(Q_1^{-1} \ldots Q_k^{-1})\Delta B_{k+1}\|_F \leq (\|Q_1^{-1}\|_2 \ldots \|Q_k^{-1}\|_2)\|\Delta B_{k+1}\|_F$$

temos

$$\|\Delta A\|_F \leq \|\Delta\tilde{A}\|_F + \prod_{i=1}^{k} \|P_i^{-1}\|_2 \|\Delta B_{k+1}\|_F.$$

Lembrando agora que $\widehat{B}_{k+1} = Q_k \cdots Q_1(A + \Delta\tilde{A})$, tem-se

$$\begin{aligned}\|\Delta A\|_F \leq & \|\Delta\tilde{A}\|_F + \prod_{i=1}^{k} \|P_i^{-1}\|_2 \left(\Gamma(\chi)\|\widehat{B}_{k+1}\|_F \prod_{i=1}^{k} \kappa_2(P_i)\right) \\ \leq & \|\Delta\tilde{A}\|_F \left(1 + \Gamma(\chi) \prod_{i=1}^{k} \kappa_2(P_i)^2\right) + \Gamma(\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i)^2 \\ \leq & \Gamma(\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i) \left(1 + \Gamma(\chi) \prod_{i=1}^{k} \kappa_2(P_i)^2 + \prod_{i=1}^{k} \kappa_2(P_i)\right) \\ \leq & \Gamma(2\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i)^3.\end{aligned}$$

□

Lema 3.11. *Seja A_{k+1} definida no lema anterior, então*

$$\|\widehat{A}_{k+1} - A_{k+1}\|_F \leq \Gamma(\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i) \left(2 + \Gamma(\chi) \prod_{i=1}^{k} \kappa_2(P_i)\right), \tag{3.22}$$

onde $\kappa_2(P_i)$ é número de condição de P_i e

$$\chi = \left(6 \sum_{i=1}^{k} \alpha_i + 4k\right) n + 10 \sum_{i=1}^{k} \alpha_i + 27k.$$

Demonstração. Como no Lema anterior seja $B_{k+1} = Q_k \cdots Q_1 A$. Então j-ésima coluna de B_{k+1} é dada por $b_j^{k+1} = Q_k \ldots Q_1 a_j$. Pelo Lema 3.9 temos

$$\widehat{b}_j^{k+1} = (Q_k + \Delta Q_k) \cdots (Q_1 + \Delta Q_1) a_j,$$

onde cada ΔQ_i, para $i = 1 : k$, satisfaz

$$\|\Delta Q_i\|_p = \|\Delta P_i\|_p \leq \Gamma\left((6\alpha_i + 4)n + 10\alpha_i + 27\right) \|P_i\|_2,$$

para $p = 2, F$. Pelo fato de que $\|P_i\|_2 \geq 1$, tem-se $\|Q_i\|_2 = \|P_i\|_2$ para todo $i = 1 : k$. Usando Lema 2.6 obtemos

$$\begin{aligned}
\|\widehat{b}_j^{k+1} - b_j^{k+1}\|_2 &= \left\| \left(\prod_{i=1}^{k} (Q_i + \Delta Q_i) a_j - \prod_{i=1}^{k} Q_i a_j \right) \right\|_2 \\
&\leq \left(\prod_{i=1}^{k} (1 + \Gamma((6\alpha_i + 4)n + 10\alpha_i + 27)) - 1 \right) \prod_{i=1}^{k} \|Q_i\|_2 \|a_j\|_2.
\end{aligned}$$

Como já foi mostrado no lema anterior,

$$\|\widehat{b}_j^{k+1} - b_j^{k+1}\|_2 \leq \Gamma(\chi) \prod_{i=1}^{k} \|P_i\|_2 \|a_j\|_2.$$

Daí

$$\|\widehat{B}_{k+1} - B_{k+1}\|_F \leq \Gamma(\chi) \prod_{i=1}^{k} \|P_i\|_2 \|A\|_F$$

Agora seja $\tilde{A}_{k+1} = \widehat{B}_{k+1}(Q_1^{-1} \cdots Q_k^{-1})$. De

$$\widehat{A}_{k+1} = \widehat{B}_{k+1}(Q_1^{-1} + \Delta Q_1^{-1}) \cdots (Q_k^{-1} + \Delta Q_k^{-1}),$$

temos

$$\begin{aligned}
\|\widehat{A}_{k+1} - \tilde{A}_{k+1}\|_F &\leq \|\widehat{B}_{k+1}\|_F \left\| \prod_{i=1}^{k} (Q_{k-i+1}^{-1} + \Delta Q_{k-i+1}^{-1}) - \prod_{i=1}^{k} Q_{k-i+1}^{-1} \right\|_F \\
&\leq \Gamma(\chi) \prod_{i=1}^{k} \|P_i^{-1}\|_2 \|\widehat{B}_{k+1}\|_F.
\end{aligned}$$

Substituindo a expressão de $\widehat{B}_{k+1}$ obtida no lema anterior temos

$$\|\widehat{A}_{k+1} - \tilde{A}_{k+1}\|_F \leq \Gamma(\chi)(\|A\|_F + \|\Delta \tilde{A}\|_F) \prod_{i=1}^{k} \kappa_2(P_i),$$

onde

$$\|\Delta\tilde{A}\|_F \leq \Gamma(\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i).$$

Por outro lado

$$\begin{aligned}\|\tilde{A}_{k+1} - A_{k+1}\|_F \leq & \|(\widehat{B}_{k+1} - B_{k+1})(Q_1^{-1} \cdots Q_k^{-1})\|_F \\ \leq & \Gamma(\chi) \prod_{i=1}^{k} \|P_i\|_2 \|A\|_F \prod_{i=1}^{k} \|Q_i^{-1}\|_2 \\ = & \Gamma(\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i).\end{aligned}$$

Portanto,

$$\begin{aligned}\|\widehat{A}_{k+1} - A_{k+1}\|_F \leq & \|\widehat{A}_{k+1} - \tilde{A}_{k+1}\|_F + \|\tilde{A}_{k+1} - A_{k+1}\|_F \\ \leq & \Gamma(\chi) \prod_{i=1}^{k} \kappa_2(P_i) \left(2\|A\|_F + \|\Delta\tilde{A}\|_F\right) \\ \leq & \Gamma(\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i) \left(2 + \Gamma(\chi) \prod_{i=1}^{k} \kappa_2(P_i)\right)\end{aligned}$$

□

Os Lemas 3.10 e 3.11 mostram o erro reverso e o erro direto da tridiagonalização no $(k+1)$-ésimo passo, respectivamente.

Toda análise acima foi feita supondo que $(x^T y)x_1 y_1 > 0$ (primeira parte do Algoritmo 3.1). Agora vamos realizar a mesma análise para o caso quando $(x^T y)x_1 y_1 < 0$.

Lema 3.12. *Sejam x, $y \in \mathbb{R}^n$ vetores tais que $(x^T y)x_1 y_1 < 0$. Então os coeficientes $\widehat{k}_1$, $\widehat{k}_2$, $\widehat{a}$ e $\widehat{b}$ e os vetores $\widehat{w}$ e $\widehat{v}$, obtidos pelo Algoritmo 3.1, são dados por*

$$\widehat{k}_1 = k_1(1 + \Theta[\alpha n + 4]), \tag{3.23}$$

$$\widehat{k}_2 = k_2(1 + \Theta[(2\alpha + 1)(n + 4)]), \tag{3.24}$$

$$\widehat{a} = -\widehat{b} = a(1 + \Theta[(2\alpha + 1)n + 4\alpha + 5]), \tag{3.25}$$

$$\widehat{w} = w(1 + \Theta[\alpha n + 5]), \tag{3.26}$$

$$\widehat{v} = v(1 + \Theta[(2\alpha + 1)n + 4\alpha + 6]), \tag{3.27}$$

onde $\alpha = \frac{|x|^T|y|}{|x^T y|}$.

Demonstração. A demonstração é feita utilizando as relações do Lema 2.5 e Lema 3.6. Para demonstrar (3.23) e (3.24) primeiro vamos analisar $\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}$. Utilizando o Lema 2.5 e a equação (3.11) temos

$$\begin{aligned}
&fl\left(\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}\right)\\
&=(1+\delta)\sqrt{|x^Ty|^2(1+\Theta[\alpha n+1])^2 + |x_1y_1||x^Ty|(1+\Theta[\alpha n+3])}\\
&=(1+\delta)\sqrt{(1+\Theta[\alpha n+3])^2(|x^Ty|^2 + |x_1y_1||x^Ty|)}\\
&=(1+\Theta[\alpha n+4])\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}.
\end{aligned}$$

Por outro lado,

$$\begin{aligned}
&fl\left(\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}\right)\\
&=(1+\delta)\sqrt{|x^Ty|^2(1+\Theta[n+1]\alpha)^2 + |x_1y_1||x^Ty|(1+\Theta[n+3]\alpha)}\\
&=(1+\delta)\sqrt{(1+\Theta[n+3]\alpha)^2(|x^Ty|^2 + |x_1y_1||x^Ty|)}\\
&=(1+\Theta[n+4]\alpha)\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}.
\end{aligned}$$

Agora o primeiro resultado utilizamos para calcular $\widehat{k}_1$,

$$\widehat{k}_1 = (1+\Theta[\alpha n+4])k_1.$$

Com segundo resultado é simples calcular a inversa, como já foi mostrado no Lema 3.7,

$$\begin{aligned}
&fl\left(\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}^{-1}\right)\\
&=\left[(1+\Theta[n+4]\alpha)\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}\right]^{-1}\\
&=(1+\Theta[(\alpha+1)(n+4)])\sqrt{|x^Ty|^2 + |x_1y_1||x^Ty|}^{-1}.
\end{aligned}$$

De onde obtemos $\widehat{k}_2$,

$$\widehat{k}_2 = ((1+\Theta[(2\alpha+1)n+4\alpha+5])k_2,$$

e os coeficientes $\widehat{a}$ e $\widehat{b}$

$$\widehat{a} = -\widehat{b} = (1+\Theta[(\alpha+1)(n+4)]).$$

Os vetores $\widehat{w}$ e $\widehat{v}$ são calculados do mesmo modo do Lema 3.8. □

As demonstrações dos três lemas a seguir são iguais aos Lemas 3.9, 3.10 e 3.11, respectivamente.

Lema 3.13. *Sejam $z \in \mathbb{R}^n$ e $y = Pz = (I + awv^T)z = z + au(v^T z)$. Então*

$$\widehat{y} = fl(z + \widehat{a}\widehat{w}(\widehat{v}^T z)) = (P + \Delta P)z,$$
$$\|\Delta P\|_p \leq \Gamma((10\alpha + 6)n + 16\alpha + 37)\|P\|_2, \quad p = 2, F. \tag{3.28}$$

onde $\widehat{a}$, $\widehat{w}$ e $\widehat{v}$ são dados pelo lema anterior.

O resultado do Lema se aplica também a $y = (P^{-1})^T y$, ou seja,

$$\widehat{y} = (P^{-1} + \Delta P^{-1})^T z, \quad \|\Delta P^{-1}\|_p \leq \Gamma((10\alpha + 6)n + 16\alpha + 37)\|P^{-1}\|_2, \tag{3.29}$$

onde $p = 2, F$.

Lema 3.14. *Considere a seqüencia de transformações*

$$A_{k+1} = Q_k A_k Q_k^{-1}, \tag{3.30}$$

onde $A_1 = A \in \mathbb{R}^{n\times n}$,

$$Q_k = \begin{pmatrix} I_k & 0 \\ 0 & P_k \end{pmatrix}, \qquad Q_k^{-1} = \begin{pmatrix} I_k & 0 \\ 0 & P_k^{-1} \end{pmatrix} \tag{3.31}$$

e $P_k, P_k^{-1} \in \mathbb{R}^{(n-k)\times(n-k)}$ são transformações de Householder generalizadas ótimas com $(x^T y)x_1 y_1 < 0$. Então

$$\widehat{A}_{k+1} = Q_k \cdots Q_1 (A + \Delta A) Q_1^{-1} \cdots Q_k^{-1} \tag{3.32}$$

onde

$$\|\Delta A\|_F \leq \Gamma(2\chi)\|A\|_F \prod_{i=1}^{k} \kappa_2(P_i)^3,$$

$\kappa_2(P_i)$ é número de condição de P_i e

$$\chi = \left(10\sum_{i=1}^{k}\alpha_i + 6k\right)n + 16\sum_{i=1}^{k}\alpha_i + 37k.$$

Lema 3.15. *Seja A_{k+1} definida no lema anterior, então*

$$\|\widehat{A}_{k+1} - A_{k+1}\|_F \leq \Gamma(\chi)\|A\|_F \prod_{i=1}^{k}\kappa_2(P_i)\left(2 + \Gamma(\chi)\prod_{i=1}^{k}\kappa_2(P_i)\right), \tag{3.33}$$

onde $\kappa_2(P_i)$ é número de condição de P_i e

$$\chi = \left(10\sum_{i=1}^{k}\alpha_i + 6k\right)n + 16\sum_{i=1}^{k}\alpha_i + 37k.$$

Observação: É razoável supor que $\Gamma(\chi)\prod_{i=1}^{k}\kappa_2(P_i) \leq 1$, logo o erro dos Lemas 3.11 3.15 pode ser dado por

$$\|\widehat{A}_{k+1} - A_{k+1}\|_F \leq 3\Gamma(\chi)\|A\|_F \prod_{i=1}^{k}\kappa_2(P_i),$$

onde χ é dado em cada um dos Lemas.

Os Lemas 3.14 e 3.15 mostram o erro reverso e o erro direto da tridiagonalização no $(k+1)$-ésimo passo, respectivamente.

Capítulo 4

Conclusão

Apresentamos no Capítulo 3 a análise de erro do algoritmo da tridiagonalização pelas transformações de Householder generalizadas ótimas. No caso particular da matriz A ser simétrica, a transformação de Householder generalizada ótima é reduzida a transformação de Housholder e, conseqüentemente, os erros da tridiagonalização devem ser iguais para os dois algoritmos descritos em Seções 2.4.1 e 3.2. De fato, lembrando que se A é simétrica então $\kappa_2(P_i) = 1$, $\alpha_i = 1$, para todo $i = 1 : k$, e os coeficientes k_1, k_2, a, b e os vetores w, v são obtidos pelo Algoritmo 3.1 para o caso quando $(x^T y)x_1 y_1 > 0$, o erro reverso e o erro direto pela transformação de Householder generalizada ótima são dados pelas Lemas 3.10 e 3.11:

$$\|\Delta A\|_F \leq \Gamma(20n^2 + 34n - 148)\|A\|_F,$$
$$\|\widehat{A}_{k+1} - A_{k+1}\|_F \leq \Gamma(20n^2 + 34n - 148)\|A\|_F,$$

enquanto os erros reverso e direto da tridiagonalização pelas transformações de Householder são dados pelos Lemas 2.9 e 2.10:

$$\|\Delta A\|_F \leq \Gamma(20n^2 + 14n - 108)\|A\|_F,$$
$$\|\widehat{A}_{n-1} - A_{n-1}\|_F \leq \Gamma(20n^2 + 14n - 108)\|A\|_F.$$

Como podemos ver a diferença entre os dois resultados, dada devido a diferença nas implementações dos Algoritmos 2.2 e 3.1, é desprezível quando n é razoavelmente grande.

Para o caso de matriz A ter todos os elementos com o mesmo sinal (essa possibilidade não é única), o erro reverso é dado pelo Lema 3.10 e o erro direto é dado pelo Lema 3.11, com $k = (n - 2)$.

No caso geral o erro reverso e o erro direto são estimados pelos Lemas 3.14 e 3.15, respectivamente, com $k = (n-2)$.

Comparando os dois erros, reverso e direto, vemos que o erro reverso depende de $\prod_i \kappa_2(P_i)^3$, enquanto o erro direto de $\prod_i \kappa_2(P_i)$. Ou seja, se as matrizes P_i forem mal condicionadas, o erro reverso é muito pior que o erro direto.

Nesse trabalho não realizamos os testes numéricos por causa da dificuldade de implementação. Para poder avaliar o erro trabalhando com precisão dupla, dada uma matriz A qualquer, precisaríamos saber qual seria a matriz tridiagonal exata obtida pelo Algoritmo 3.2, sem os erros numéricos. Uma solução seria implementar os Algoritmos 3.1 e 3.2 em duas precisões diferentes, com condição de que a mantissa de uma precisão fosse o dobro da outra, por exemplo precisão dupla e precisão dupla estendida. Então, a matriz tridiagonal resultante, calculada na precisão dupla estendida, seria igual a matriz tridiagonal exata arredondada para precisão dupla (Seção 2.3.1). Mas nos computadores pessoais (baseados nos processadores com arquitetura x86) a mantissa da precisão dupla possui 53 bits e a mantissa da precisão dupla estendida apenas 64 bits, o que torna inviável essa comparação. Existem supercomputadores nos quais a precisão dupla estendida possui a mantissa com 105 bits. Mas atualmente não temos acesso a nenhuma dessas máquinas.

Apêndice A

Padrão IEEE-754

Quando se quer falar de um sistema de ponto flutuante ou, quando se quer caracterizá-lo, é necessário especificar: a base numérica do sistema; o número de dígitos da mantissa; o intervalo de abrangência do expoente da base e como é feita a representação destes números; como é feito o tratamento de underflow e overflow; como são efetuados as operações aritméticas e os tipos de arredondamentos disponíveis e utilizados nas operações, pois isto influenciará a análise e quantificação dos erros que cálculos efetuados neste sistema terão.

A.1 Objetivos e especificações

Em 1985 o IEEE (Institute of Electral and Eletronics Engineers) publicou o padrão 754 [7], que define a aritmética binária de ponto flutuante e como tratar os casos especiais que ocorrem durante a resolução de um determinado cálculo ou problema.

O padrão IEEE-754 especifica:

1. formato do número de ponto flutuante estendido e simples;
2. operações de adição, subtração, multiplicação, divisão, raiz quadrada, resto e operações de comparação;
3. conversão entre inteiros e formatos de ponto flutuante;
4. conversão entre diferentes formatos de ponto flutuante;
5. conversão entre números em ponto flutuante simples e strings decimais;

6. exceções em ponto flutuante e suas manipulações, incluindo "não-número"(NaNs).

A.2 Definições

- **"Biased"expoente**: a soma de uma constante (bias) ao expoente, de forma que o expoente não seja nunca negativo, facilitando a representação. É uma mudança de escala. Ex: Cray.

- Número binário em ponto flutuante: um cadeia de bits é caracterizado por três componentes. São eles um sinal, um expoente sinalizado, uma mantissa. O valor numérico, se existir, é o produto sinalizado da mantissa e da base binária elevado à potência do expoente. Neste padrão nem sempre é possível distinguir um cadeia de bits do número que ela representa.

- **Número desnormalizado**: um número não nulo em ponto flutuante cujo expoente tem um valor reservado, usualmente o formato mínimo e cujo o primeiro bit da mantissa é zero. É útil para representar números menores do que o menor número de máquina do padrão normalizado.

- **Destino**: a posição do resultado de uma operação binária ou unária. O destino pode ser explicitamente especificado pelo usuário ou implicitamente suprido pelo sistema (por exemplo, resultados intermediários em sub-expressões ou argumentos de procedimentos). Algumas linguagens colocam os resultados intermediários de cálculos em destinos fora do controle do usuário. Este padrão define o resultado de uma operação em termos do formato do destino e dos valores dos operandos.

- **Expoente**: o número 2 é elevado a este componente do número binário de ponto flutuante. Ocasionalmente, o expoente é chamado de "expoente sinalizado"ou "un-biased expoente".

- **Fração**: o campo da mantissa que fica do lado direito do ponto binário implícito.

- **Modo**: uma variável que um usuário pode marcar, salvar e restaurar para controle de execução de uma operação aritmética subseqüente. O modo "default"é aquele que está ativo no programa, a menos que uma declaração contrária explícita seja incluída no programa ou em suas especificações. O seguinte modo deve ser implementado: arredondamento, para controlar a direção dos erros de arredondamento.

Em certas implementações, a precisão do arredondamento pode ser necessária, para encurtar a precisão do resultado. O implementador pode, se quiser, implementar os seguintes modos: "traps"habilitados/desabilitados, para o tratamento de exceções.

- **NaNs**: não-número. É um código de uma entidade simbólica no formato de ponto flutuante. Sua finalidade é não interromper o cálculo em situações tais como 0/0 e raiz de um número negativo, permitindo que a computação continue, mas indicando que houve algum cálculo inválido. Existem dois tipos de NaNs:

  a) NaNs sinalizados: indicam exceção de operação inválida, quando as operações aparecem como operandos.

  b) NaNs quietos: se propagam através de quase todas as operações aritméticas, sem sinalizar exceções.

- **Resultado**: é uma cadeia de bits (geralmente representando um número) que é entregue a um destino.

- **Mantissa**: um componente de um número binário em ponto flutuante que consiste de um explícito ou implícito primeiro bit à esquerda do ponto e o campo de fração à direita.

- **Deve**: quando esta palavra aparecer no texto significa que o que está sendo referido deve constar em qualquer implementação.

- **Deveria**: o uso desta palavra "deveria"significa que é fortemente recomendado, ou seja, está mais de acordo com o padrão; porém a arquitetura da máquina ou outros impedimentos podem tornar impraticável a implementação conforme o padrão sugere.

- **Flags de estados**: uma variável que pode tomar dois estados, ligado/desligado. O usuário pode desligar o flag, copiar, restaurar o estado anterior. Quando ligado, o estado do flag pode conter informações adicionais do sistema, possivelmente inacessíveis a alguns dos usuários. As operações desse padrão podem ligar, como um efeito colateral, alguns dos seguintes flags:

  a) resultado inexato

  b) underflow

c) overflow

d) divisão por zero

e) operação inválida

- **Usuário**: qualquer pessoa, hardware ou programa não especificado pelo padrão, e que tenha acesso e controle às operações do ambiente de programação especificado neste padrão.

A.3 Formatos

Esse padrão define 4 formatos de ponto flutuante divididos em dois grupos, simples e estendido, cuja precisão é simples e dupla. O nível de implementação deste padrão é caracterizado pela combinação dos formatos suportados.

A.4 Conjunto de Valores

Esta seção diz respeito somente à representação dos valores numéricos representados dentro do formato, não à codificação. Os únicos valores representáveis num determinado formato são aqueles especificados pelos três parâmetros inteiros que seguem:

p - número de bits da mantissa (precisão),

e_{max} - expoente máximo e

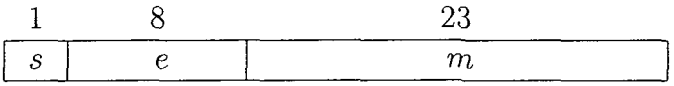
e_{min} - expoente mínimo.

A tabela a seguir resume os parâmetros dos formatos de ponto flutuante.

Parâmetro	Formato			
	Simples	Simples Estendido	Duplo	Dupla Estendido
p	24	≥ 32	53	≥ 64
e_{max}	$+127$	$\geq +1023$	$+1023$	$\geq +16383$
e_{min}	-126	≤ -1022	-1022	≤ -16382
bits do expoente	8	≥ 11	11	≥ 15
total em bits	32	≥ 43	64	≥ 79

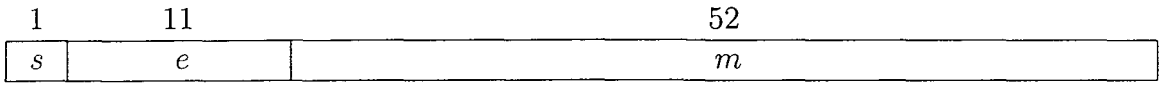
A.5 Formato simples

O formato simples é constituído de 32 bits, distribuídos como mostra a figura



A.6 Formato duplo

O formato duplo é constituído de 64 bits, como mostra a figura



A.7 Formato estendido

Os formatos simples estendido e duplo estendido são codificados de acordo com as restrições da tabela acima. Este padrão permite na implementação codificar alguns valores redundantemente. Esta redundância tem que ser transparente para o usuário nos seguintes aspectos: ou a implementação codifica cada valor não nulo de uma forma única, ou interpreta os valores não nulos redundantes como um mesmo valor. Uma implementação pode também reservar alguns bits para propósitos que vão além do escopo deste padrão. Quando esses bits reservados aparecem como operandos o resultado não é especificado. Numa implementação não deve ser assumido que o formato simples estendido possua um intervalo maior do que o formato duplo.

A.8 Combinação de formatos

Toda a implementação conforme este padrão deverá suportar o formato simples. Uma implementação deveria suportar o formato estendido correspondente ao formato básico suportado, não necessitando suportar qualquer outro formato estendido.

A.9 Arredondamento

O arredondamento toma um número com precisão infinita e, se necessário, o modifica para poder coloca-lo no destino, sinalizando a exceção de inexatidão. Exceto para conversão binário-decimal, todas as operações especificadas acima devem ser executadas como se o primeiro resultado intermediário correto tivesse precisão infinita e intervalo ilimitado e então o resultado é arredondado de acordo com um dos quatro modos de arredondamento (arredondamento para o número mais próximo de máquina, arredondamento para $+\infty$, arredondamento para $-\infty$ e arredondamento para 0).Os modos de arredondamento afetam todas as operações aritméticas com exceção da comparação e do resto. Os modos de arredondamento podem afetar o sinal da soma de zeros. Quando ocorre o arredondamento e se está muito perto de situações de overflow e underflow essas situações podem ser sinalizadas.

A.10 Operações

Todas as implementações que estão de acordo com este padrão devem prover operações de:

- adição, subtração, multiplicação, divisão,
- extração da raiz quadrada,
- encontrar o resto,
- passar de/para número em formato de ponto flutuante para formato inteiro,
- conversão entre diferentes formatos de ponto flutuante,
- arredondar de número em formato de ponto flutuante para inteiro,
- conversão binária–decimal,
- comparação.

É uma opção de implementação considerar a cópia sem mudança de formato como uma operação. Exceto para conversão binária–decimal, todas as operações devem ser

feitas considerando o primeiro resultado correto intermediário com precisão infinita e intervalo ilimitado, e então adequar este resultado intermediário no formato do destino.

Apêndice B

Normas

B.1 Normas vetoriais

Uma norma vetorial em $\mathbb{R}^n$ é uma função $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ que satisfaz as seguintes propriedades:

1. $\|x\| \geq 0$ para todo $x \in \mathbb{R}^n$, e $\|x\| = 0 \Leftrightarrow x = 0$,
2. $\|\alpha x\| = |\alpha|\|x\|$ para todo $\alpha \in \mathbb{R}$, $x \in \mathbb{R}^n$,
3. $\|x + y\| \leq \|x\| + \|y\|$ para todo $x, y \in \mathbb{R}^n$.

Uma classe usual de normas vetoriais são as p-normas (ou normas de Hölder), definidas por

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}, \qquad p \geq 1.$$

Dentro desses normas as mais mais importantes em análise de erro são

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|,$$

$$\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2} = (x^T x)^{1/2},$$

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|.$$

B.2 Algumas propriedades das normas vetoriais

Um resultado clássico envolvendo p-normas é a *desigualdade de Hölder*:

$$|x^T y| \leq \|x\|_p \|y\|_q, \qquad \frac{1}{p} + \frac{1}{q} = 1. \tag{B.1}$$

Um caso especial muito importante decorrente de (B.1) é *desigualdade de Cauchy-Schwartz*:

$$|x^T y| \leq \|x\|_2 \|y\|_2. \tag{B.2}$$

Todas as normas em $\mathbb{R}^n$ são *equivalentes*, isto é, se $\|\cdot\|_\alpha$ e $\|\cdot\|_\beta$ são normas em $\mathbb{R}^n$, então existem constantes positivas a e b tais que

$$a\|x\|_\alpha \leq \|x\|_\beta \leq b\|x\|_\alpha$$

para todo $x \in \mathbb{R}^n$.

Para normas 1,2 e ∞, se $x \in \mathbb{R}^n$, temos as seguintes relações:

$$\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\|x\|_2, \tag{B.3}$$

$$\|x\|_\infty \leq \|x\|_2 \leq \sqrt{n}\|x\|_\infty, \tag{B.4}$$

$$\|x\|_\infty \leq \|x\|_1 \leq n\|x\|_\infty. \tag{B.5}$$

Veja, por exemplo, [3] ou [6].

B.3 Normas matriciais

Uma norma matricial é uma função $\|\cdot\| : \mathbb{R}^{m\times n} \to \mathbb{R}$ que satisfaz as seguintes propriedades:

1. $\|A\| \geq 0$ para todo $A \in \mathbb{R}^{m\times n}$, e $\|A\| = 0 \Leftrightarrow A = 0$.
2. $\|\alpha A\| = |\alpha|\|A\|$ para todo $\alpha \in \mathbb{R}$, $A \in \mathbb{R}^{m\times n}$.
3. $\|A + B\| \leq \|A\| + \|B\|$ para todo $A, B \in \mathbb{R}^{m\times n}$.

As normas matriciais mais usadas em análise de erro são as p-normas

$$\|A\|_p = \sup_{x\neq 0} \frac{\|Ax\|_p}{\|x\|_p} \tag{B.6}$$

e a norma de Frobenius

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}. \tag{B.7}$$

B.4 Algumas propriedades das normas matriciais

Dizemos que a norma $\|\cdot\|$ é *consistente* se para quaisquer matrizes $A \in \mathbb{R}^{m\times n}$ e $B \in \mathbb{R}^{n\times q}$ tem-se

$$\|AB\| \leq \|A\|\|B\|.$$

A norma de Frobenius e as p-normas (especialmente $p = 1, 2, \infty$) satisfazem algumas desigualdades importantes expostas a seguir. Seja $A \in \mathbb{R}^{m\times n}$ então tem-se

$$\|A\|_1 = \max_{1\leq j\leq n}\sum_{i=1}^{n}|a_{ij}|, \tag{B.8}$$

$$\|A\|_\infty = \max_{1\leq i\leq n}\sum_{j=1}^{n}|a_{ij}|, \tag{B.9}$$

$$\|A\|_2 \leq \|A\|_F \leq \sqrt{\text{posto}(A)}\|A\|_2, \tag{B.10}$$

$$\max_{i,j}|a_{ij}| \leq \|A\|_2 \leq \sqrt{mn}\max_{i,j}|a_{ij}|, \tag{B.11}$$

$$\frac{1}{\sqrt{n}}\|A\|_\infty \leq \|A\|_2 \leq \sqrt{m}\|A\|_\infty, \tag{B.12}$$

$$\frac{1}{\sqrt{m}}\|A\|_1 \leq \|A\|_2 \leq \sqrt{n}\|A\|_1. \tag{B.13}$$

Ver [3] ou [6].

Referências

[1] N. S. Bahvalov, N. P. Zhidkov, and G. M. Kobel'kov. *Chislennye metody: Uchebnoe posobie*. Nauka, Moskva, 1987. [em russo].

[2] D. Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, 23(1):5–48, 1991.

[3] G.H. Golub and C. Von Loan. *Matrix Computation*. The Johns Hopkins University Press, Baltimore and London, 3rd edition edition, 1996.

[4] G.H. Golub, J.Y. Yuan, R. Biloti, and J.D.P. Ramos. Computation of eigensystems by tri-diagonalization. Technical report, Departamento de Matemática, Universidade Federal do Paraná, 2003.

[5] N.J. Higham. The accuracy of floating point summation. *SIAM J. Sci. Comput.*, 14(4):783–799, 1993.

[6] N.J. Higham. *Accurancy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, 2nd edition edition, 2002.

[7] Institute of Electrical and Electronics Engineers. *IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Standard 754-1985*, 1985.

[8] Intel Corp., Mt. Prospect, Illinois. *IA-32 Intel Architecture Software Developer's Manual*, 2001.

[9] D.E. Knuth. *The Art of Computer Programming, Volume 2, Seminumerical Algorithms*. Addison-Wesley, Reading, MA, USA, 3rd edition edition, 1998.

[10] C.D. LaBudde. The reduction of an arbitrary real square matrix to tridiagonal form using similarity transformations. *Math. Comp.*, 17:433–437, 1963.

[11] D. Morgan. *Numerical Methods/Real-Time and Embedded Systems Programming.* M&T Publishing, 1992.

[12] J. H. Wilkinson. *Rounding Errors in Algebraic Processes.* Prentice-Hall, 1963.

[13] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, London, 1965.